(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 August 2001 (16.08.2001)

PCT

(10) International Publication Number
WO 01/59661 A2

(51) International Patent Classification⁷: G06F 17/60

(21) International Application Number: PCT/US01/03987

(22) International Filing Date: 8 February 2001 (08.02.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/500,322 8 February 2000 (08.02.2000) US

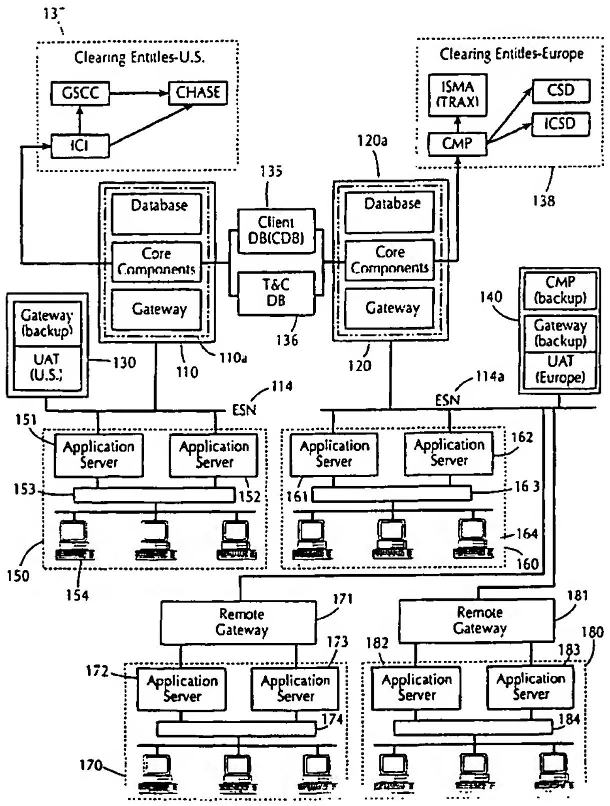
(71) Applicants and
(72) Inventors: **FENICHEL, Peter, I.** [GB/GB]; 23 Maragetta Terrace, London SW3 5NU (GB). **EZRA, Michael, J.** [GB/GB]; 14 Arkwright Road, Hampstead, London NW3 6BG (GB). **CARY, Steven, H.** [US/US]; 180 Pear Tree Point Road, Darien, CT 06820 (US). **JOHNSON-WATT, Duncan** [GB/GB]; 82 Warwick Park, Tunbridge Wells, Kent TN2 5EF (GB). **LEFFERTS, David, H.** [US/US]; Apartment 7A, 420 East 54th Street, New York, NY 10022 (US). **MISCIMARRA, Anthony, F., Jr.** [US/US]; 9 George Langeloh Court, Rye, NY 10580 (US). **MOSELEY, Rachel, N.** [US/US]; One Berwick Road, Scarsdale, NY 10583 (US). **PAULL, Michael, John, Crichton** [GB/GB]; 1 Kersley Mews, Battersea, London SW11 4PS (GB). **ROSS, Howard, L.** [US/US]; 92 Irma Drive, Oceanside, NY 11572 (US).

(74) Agents: **SHARROTT, Douglas** et al.; Fitzpatrick, Cella, Harper & Scinto, 30 Rockfeller Plaza, New York, NY 10112 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ,

[Continued on next page]

(54) Title: APPARATUS, METHOD AND PROGRAM FOR A FIXED INCOME TRADING SYSTEM



(57) Abstract: A trading apparatus includes a network server. The network server includes at least one processor which executes computer executable code, an interface through which orders are received by the trading apparatus, and a memory. The memory stores computer executable code, the code (i) maintains a database pertaining to trading activity; (ii) processes passive orders; (iii) processes aggressive orders; and (iv) executes trades based on the aggressive orders.

WO 01/59661 A2

NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) **Designated States** *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

TITLE OF THE INVENTION

APPARATUS, METHOD AND PROGRAM FOR A FIXED INCOME TRADING SYSTEM

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates generally to a method and apparatus for trading securities, and in particular, to trading fixed income instruments through a computerized system.

Related Background Art

Perhaps the oldest mechanism for facilitating the trading of financial instruments is the market concept, in which buyers and sellers (or their agents) collect at a single location and negotiate the price at which a purchase or sale occurs.

Markets can be highly efficient for both buyers and sellers. Some exchange models have an organization that provides a floor where dealers can engage in buy and sell transactions, either directly or through specialists who maintain liquidity in particular instruments.

Exchange models are used for the purchase and sale of relatively fungible items, such as equity financial instruments, which have characteristics promoting liquidity: they are widely held, frequently traded and have a relatively large number of shares outstanding. However, exchanges have not been extensively used for buying and selling fixed income instruments, due to the complex issues surrounding fixed instruments, including the typically large monetary transaction sizes.

The advent of computer technology is having a major impact on the practice on the buying and selling of financial instruments. In particular, computer systems have been implemented to facilitate securities trading in a variety of ways. A recent example of such a system is described in U.S. Patent No. 5,924,082, which allows buyers and sellers located at remote computer terminals to engage in buy/sell transactions. The system automatically matches potential parties to a transaction, based upon ranking information, and then permits the potential parties to electronically negotiate some or all terms of the transaction.

The use of computerized systems is becoming widespread for the trading of equity instruments. However, the development of a computerized trading system for fixed income products, which takes full advantage of the potential of modern data processing systems to reduce market costs, has yet to be realized.

SUMMARY OF THE INVENTION

The inventors have developed a fixed income trading system that provides traders (defined as persons or entities that buy and/or sell securities) with a computerized trading venue. Fixed instruments are entered into the fixed instrument trading system and are organized into books for system wide display. A book is defined by a set of bids and/or offers regarding an instrument and by a settlement date. Traders enter "firm" orders (e.g., offers and bids that can be immediately executed against at a stated price) into the respective instrument books, which are then displayed system wide. An "offer" can be defined as an order to sell, and a "bid" can be defined as an order to buy. A trader can monitor orders and transactions through a graphic user interface ("GUI") to the system. Traders can "aggress" an order (e.g., hit a bid, or lift an offer) through the GUI and system. A hit can be defined as an acceptance of a bid (e.g., a trader "hitting" a bid is willing to sell at a bid price or higher). A lift can be defined as an acceptance of an offer (e.g., a trader "lifting" an order is willing to buy at the offer price, or lower). A successfully hit bid, or a lifted offer, results in an executed trade. Executed trades can be automatically forwarded to a clearing entity for settlement. Settlement can be defined as a process where a first trader exchanges currency or value for a second trader's security.

The present invention significantly reduces transactional costs associated with disseminating information regarding fixed income instruments. Furthermore, the present invention simplifies instrument trading by organizing and centralizing instrument information. The present invention also provides redundant backup features to ensure system vitality.

Accordingly, one aspect of the present invention is to provide a trading system for trading fixed income instruments. The system includes at least two server computers. Each server computer includes a first interface to communicate with a server computer; a second interface through which orders are received by the system; at least one processor for executing computer code; a first database pertaining to trading activity through the system in a first trading market; a second database pertaining to trading activity through the system in a second trading market; at least one cache storing information regarding market activity in the respective first or second market; and a memory having computer executable code stored thereon. The code is for (i) updating the first and second databases and the at least one cache; (ii) synchronizing the updates; and (ii) processing orders from each respective trading market.

Another aspect of the present invention is to provide a trading system, wherein each of two server computers is located in a geographically separate location. The trading system, further includes at least one trader site including at least one application server and a workstation connected to the at least one application server. The at least one application server contains a cache having at least a copy of the information in the first database stored thereon. The system includes computer executable code that i) updates the cache in the at least one application server, and ii) synchronizes the updates with the updates to the first and second databases.

Another object of the present invention is to provide a trading system having computer executable code stored in a memory, wherein the code maintains at least one instrument book defined by a set of bids and offers and by a settlement date. The code changes a status of the at least one instrument book. A status includes open, closed, inactive and retired.

Another object of the present invention is to provide a trading system having code stored in a memory, the code ranks a bid having a highest price first among other bids in at least one book. The code also ranks an offer having the lowest price first among other offers in the at least one book. The code also ranks passive orders having a same price in the at least one book on a first come, first serve basis with respect to each other. The code processes aggressive orders on a first come, first serve basis, and removes from the at least one book a passive bid or passive offer once the passive bid or passive offer has been executed. The code also validates orders.

Another object of the present invention is to provide a trading system with a user terminal interfaced with the apparatus. Preferably, at least one book is configured for display through the user terminal. The trading system contains code to filter data transmitted to the user terminal.

Yet another object is to provide a trading system with code to cause the system to transfer information related to executed transactions through the trading system to at least one clearing entity.

Still another object of the present invention is to provide a trading apparatus. The apparatus includes at least one network server having: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) processing a sweep order for the at least one instrument book.

According to one aspect, the sweep order is an order to aggress multiple passive orders at a selected price option and amount. The code aggresses passive orders to fill the sweep order by sequentially i) executing shown amounts of passive orders at the selected price; ii) executing reserve amounts at the selected price when needed to fill the sweep order; and iii) executing passive orders at a next price when needed to fill the sweep order. The code expires a first passive order listed in the at least one instrument book when i) the code determines that the first passive order was submitted by a trader who also submitted the sweep order, and ii) the code executes a second passive order that is listed below the first passive order.

Yet another object of the present invention is to provide a trading apparatus that receives a first passive order that includes price and amount information. The first passive order further includes shown amount information, and the code determines a reserve amount from the shown amount. At least one user terminal is interfaced with the trading apparatus and the code conditionally conceals the reserve amount from the at least one user terminal. The codes conveys the reserve amount to the at least one user terminal when a condition includes a user who entered the first passive order. The code also executes a shown amount prior to executing a reserve amount. The code also executes shown amounts for orders at a same price in an instrument book before executing reserve amounts for the orders at the same price.

Still another object of the present invention is to provide a trading apparatus including at least one network server. The at least one network server includes: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders. The steps process aggressive orders each having instrument identification, price option information, amount information and minimum amount for execution information.

According to yet another aspect, the trading apparatus includes code to execute a trade only when a total amount of shown and reserve amounts for orders at the order price option in at least one instrument book equals at least the minimum amount for execution. Preferably, the code generates an execution notification regarding the order only when a trade is executed.

Still another object of the present invention is to provide a trading apparatus having at least one network server, including: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; and (iv) executing trades based on the aggressive orders. The steps process aggressive orders each having instrument identification, price option criteria, amount information, and a post residual authorization.

According to one aspect, a trading apparatus includes code stored in a memory that converts an unfilled balance of the first aggressive order into a passive order having an aggressed-on price. The code posts the unfilled amount in at least one instrument book as a shown amount. Alternatively, the code posts a portion of the unfilled amount in the at least one instrument book as a shown amount and the remainder as a reserve amount.

Still another object is to provide a trading apparatus including at least one network server having: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) suspending all orders submitted by a user through a user terminal when communication with the user terminal is interrupted.

Another object of the present invention is to provide a trading apparatus with at least one network server, including: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) canceling all orders received from a user through a user terminal in response to a user generated bulk cancel request.

Another object of the present invention is to provide a trading apparatus having at least one network server, including: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) suspending all orders received from a user through a user terminal in response to a user generated bulk suspend request.

Yet another object of the present invention is to provide a trading apparatus having at least one network server, including: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) converting passive orders into an aggressive orders and executing the converted aggressive orders at a top of book price, when the passive orders invert a market. A first passive order inverts the market when the first passive order includes a bid having a higher price than the lowest priced offer in the at least one instrument book, or when the first passive order includes an offer having a lower price than the highest priced bid in the at least one instrument book.

Yet another object of the present invention is to provide a trading apparatus having at least one network server including: at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) expiring all orders at the end of a trading day.

Another object of the invention is to provide a trading apparatus having at least one network server, including at least one processor which executes computer executable code; an interface through which orders are received by the trading apparatus; and a memory for storing computer executable code. The code includes steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) expiring orders based on user selected criteria.

Another object of the present invention is to provide media containing computer executable code. The code is to generate a graphic user interface invocable by programable software for use in a fixed income trading system. The system provides a computerized venue for trading instruments. The system receives orders including bids and offers for individual instruments submitted to the system. The system maintains at least one book defined by a set of orders and by a settlement date. The graphic user interface is for employment on a trader work station connected to the system. The computer executable code includes steps to generate a first window having a first display region displaying a top of book view, a second display region displaying a key issues view, a third region displaying a depth of book view, and a forth region including at least one user activatible button to invoke an order entry dialog box through which a user inputs order requests into the system. The computer executable code further includes steps to generate a region including a user activatible button for displaying a floating book window. The floating book includes a depth of book view and a region including user activatible buttons to invoke at least one order entry dialog box.

In still another embodiment, media containing computer executable code to generate a graphic user interface invocable by a software program is provided. The graphic user interface for use by brokers in a trading system. The trading system provides a computerized venue for trading instruments. The code includes steps to generate a first window displaying a scrolling list of real-time market activity that occurs through the system. The first window including a first region having a pull-down menu for a user activatable filter for selecting a subset of activity to view for real-time display. The code includes steps to generate a second window displaying a client-specific blotter including order history information, order status information, and user activated transactions, wherein a broker accessing the second window can interface with the system on behalf of the specified client.

In another embodiment, a network trading system is provided. The system includes at least one server including at least one processor for executing computer code, an interface through which a user gains access to the system, and at least one memory having computer executable code stored thereon. The code: (i) tracks activity through the system with respect to individual trading instruments; (ii) organizes for real-time display the tracked activity of individual trading instruments; and (iii) displays to the user a subset of the tracked activity.

These and other objects, features and advantages will be apparent from the following description of the preferred embodiments of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a diagram illustrating a system overview of a fixed income trading system.

Figure 2a is a diagram showing a three-domain partitioning configuration of a network computer server, and Figure 2b is a diagram showing an embodiment having two network servers per each data center of Figure 1.

Figure 3 is a diagram illustrating individual components within a network server.

Figure 4 is a diagram showing a communication scheme within the core component domain, including a data synchronization data stream and communication involving an AQP core component.

Figure 5 is a diagram illustrating a retransmission request from a consuming component.

Figure 6 is a diagram illustrating heartbeats issued from the AQP core component.

Figure 7 is a diagram illustrating the redundancy features of a DR Replicator core component.

Figure 8 is a diagram illustrating MCP instances in the core and gateway domains.

Figure 9 is a diagram illustrating a possible domain configuration for a data center of Figure 2b.

Figure 10 is a diagram illustrating a cluster management channel, and a failover procedure involving the MCP.

Figures 11a-11d are diagrams illustrating how instrument data is supplied to the system of Figure 1.

Figures 12a-12c are diagrams illustrating how client data is supplied to the system of Figure 1 via a client database adapter core component.

Figures 13a-13e illustrate a function of a Back Office Adapter component operating within the system illustrated in Figure 1.

Figures 14a-14h illustrate a procedure for entering and executing orders through the system illustrated in Figure 1.

Figure 15 is a diagram illustrating network connections for the system illustrated in Figure 1.

Figure 16 is a diagram illustrating a transaction in a U.S. market through the system illustrated in Figure 1, between a Firm A and a Firm B.

Figures 17a and 17b are diagrams showing an interaction between the system of Figure 1 and outside clearing entities for U.S. markets.

Figures 18a-18c are diagrams illustrating a component commit operation, which can change an object's status.

Figure 19 is a diagram illustrating how a trade occurs through the system of Figure 1.

Figure 20 is a diagram illustrating a settlement process in a European market.

Figure 21 is a diagram illustrating how a TRAX system reports matching results back to traders in a European market.

Figure 22 is a diagram illustrating a settlement confirmation in a European market.

Figure 23 is a screen shot showing a Login dialog box of a trader GUI.

Figure 24 is a screen shot showing a Main Market view of the trader GUI.

Figure 25 is a screen shot showing an Order Entry dialog box of the trader GUI.

Figure 26 is a screen shot showing a Hit Bid dialog box of the trader GUI.

Figure 27 is a screen shot showing a Market pull-down for the trader GUI.

Figure 28 is a screen shot showing a Preferences dialog box of the trader GUI.

Figure 29 is a screen shot showing an Offer Order entry box of the trader GUI.

Figure 30 is a screen shot showing a Take Offer dialog box of the trader GUI.

Figure 31 is a diagram illustrating an execution priority for given shown and reserve amounts in an instrument book.

Figure 32 is a screen shot showing a Sweeping the Book dialog box of the trader GUI.

Figure 33 is a screen shot showing a Quick/Hit dialog box of the trader GUI.

Figure 34 is a screen shot showing a Quick/Take dialog box of the trader GUI.

Figure 35 is a flow diagram illustrating one aspect of an auto-aggress feature of the present invention.

Figures 36a-36c are screen shots showing a Blotter of the trader GUI.

Figure 37 is a screen shot showing a Floating Book of the trader GUI.

Figures 38a and 38b are screen shots showing Cancel Orders and Suspend Orders dialog boxes, respectively.

Figure 39 is a screen shot showing an Execution Notification window of the trader GUI.

Figure 40 is a screen shot showing a window of the trader GUI having multiple Depth of Book views.

Figure 41 is a screen shot of a broker GUI broker desk tool bar.

Figure 42 is a screen shot of the broker GUI showing a real-time market monitor application.

Figure 43 is a screen shot of the broker GUI showing a Create Market Monitor Filter dialog box for orders/trades.

Figure 44 is a screen shot of the broker GUI showing a Create Market Monitor Filter dialog box for instruments.

Figure 45 is a screen shot of the broker GUI showing a drop-down menu of instrument groupings.

Figure 46 is a screen shot of the broker GUI showing a selected group according to Figure 45.

Figure 47 is a screen shot of the broker GUI showing a client listing.

Figure 48 is a screen shot of the broker GUI showing a list of trades.

Figure 49 is a screen shot of the broker GUI showing a main market view.

Figure 50 is a screen shot of the broker GUI showing a suspend/cancel feature of the present invention.

Detailed Description of the Preferred Embodiments

Figure 1 illustrates in schematic form the principal components utilized in the present invention. In particular, there is shown a fixed income trading system 100, which facilitates trading of fixed income instruments in respective trading markets.

Overview of Capabilities of System 100

From a trader's perspective, the system 100 allows users (e.g., "traders") to enter order requests that are displayed network-wide as firm bids to buy or offers to sell at a price/quantity for a particular fixed income instrument. All orders are preferably live; meaning, that they can be immediately executed against at a listed price. Traders can be remotely located at trading desks (e.g., user terminals) throughout the world, and communicate by means of desktop workstations or other information appliances that have the ability to communicate with the balance of system 100.

System 100 allows traders or client users to manage orders (or a subset of orders), including tasks such as modifying, canceling, or suspending, or reinstating previously suspended orders, and allows a user to accept (e.g., "aggress") bids/offers, which if successful result in a trade. The system 100 further allows control of the amount of an order that is held in reserve and therefore not displayed publicly on the system. The term "reserve" refers to the portion of an order that is available for execution, but not displayed to other traders. Reserve functionality helps traders avoid moving the market against them by not disclosing to other traders the total amount of their orders. In a preferred embodiment for U.S. markets, the system does not indicate that a reserve amount exists for orders to anyone other than the order's owner. In a preferred embodiment for European markets, the system (through a trader GUI) indicates an "R" if an order has a reserve amount.

System 100 arranges offers and bids into books. A "book" is defined as a set of bids and/or offers that compete with each other for the same instrument and settlement date in the same market type (e.g., "type" refers to cash or basis trading). If bids or offers exist for multiple settlement dates for a single security, they are preferably maintained in separate books.

System 100 preferably allows a trader to accept a single order, or multiple orders simultaneously. Simultaneous acceptance is referred to as "sweeping the book." System 100 sweeps the book by beginning with the best bid/offer (from the aggressor's standpoint) and accepting that order entirely, including reserve amounts, before moving on to the next order at a next price.

A further preferred aspect of the system 100 is an organization and display feature that allows a user to view trading activity on the system 100. A trader can access several different views of a book, including the "Top of Book," the "Depth of Book," and "Key Issues." The "Top of Book" view includes information about the current best bid/offer for securities on the system 100. The "Depth of Book" view includes real-time information about a security's active orders on the system 100. The "Key Issues" view provides information which allows a user to create a customized view of the top of the book activity for a specific set of securities. Furthermore, the system 100 allows a user to maintain a "blotter" of activity. The blotter is a real-time work space that enables users to monitor and manage all relevant details about their orders and trades.

In addition to the above-mentioned advantages, the system 100 includes many other preferred functional features. For example, the system 100 preferably has a user authorization and permission protocol. These protocols provide user-based restrictions on selected system functions. For example, the system 100 can verify whether a user has clearance to access the system. This verification can be facilitated through system generated ID codes, electronic signatures or through terminal passwords, for example. An order validation function is also provided to ensure that orders follow established "trade execution rules," as well as ensuring users are advised of entry mistakes. Price/time priority is a trade execution rule that specifies how orders are sequenced for display and execution on the system 100. From a display perspective, currency bids are sequenced from highest to lowest price. Yield and discount bids are sequenced from lowest to highest price. Currency offers are sequenced from lowest to highest. Yield and Discount offers are sequenced from highest to lowest price. Within a single price, orders are sequenced oldest to newest. From an execution perspective, the trade execution rules preferably prioritize executions as follows: i) the system 100 finds the best price, and the best price executes first, and ii) if multiple orders at a given price exist, the system takes the oldest best price first (e.g., the price associated with the order that has been on the system longer than any other order at that price).

The system also preferably includes order entry functions such as various pricing options, reserve/display capabilities, order modification, cancellations, and reinstatement capabilities, as are discussed below. Additionally, a trade execution function is provided to support bid/offer options. Also provided is a data filtering function to ensure that trader "blotter" views display only the information that is related to a given trader's market activity. Furthermore, the system provides real-time trade updates to all trader desktops system wide through a diverse range of processing functions such as object creation/management, data persistence and performance techniques. Other functions preferably include the creation of standard market level displays, a resiliency function to ensure data input from any trader desktop is not lost in the event of a hardware or software failure, commission calculation functions, support for clearing and settlement by providing links to external clearing facilities, and Broker Desk tools (herein after "B-Desk tools") to enable brokers and system personnel to support system activity. A broker can be defined as a party that is neutral with respect to a trade. The B-Desk tools provide brokers with the ability to view orders and transaction data, manage client information, set up book parameters, and maintain instrument information, for example.

General Description of System 100

In Figure 1, trader desktops (e.g., workstations) 154, 164, 175 and 188 are arranged to provide traders with an access point to system 100. A workstation is a computer preferably having a memory (e.g., RAM, ROM, and/or local fixed storage), at least one processor (e.g., a CPU), a monitor, and data entry devices (e.g., a key board, mouse, light pen, touch sensitive monitor, etc.). The workstations are controlled by an operating system, such as Windows 98, from Microsoft Corporation. As will be appreciated by those skilled in the art, an equivalent system such as UNIX, could be used. Software is installed (e.g., stored in memory) on each workstation that produces a trader Graphic User Interface ("GUI"), an example of which is shown in Figure 24. As will be appreciated by those skilled in the art, the software can be stored on computer readable media, including CD-ROMs, floppy disks, removable media, magneto-optical media, fixed disks, flash memory, memory sticks, etc. The trader GUI will be discussed in greater detail below.

Workstations 154 are preferably connected through a known firewall 153 to application servers 151 and 152. The application servers 151 and 152 provide object management processing, data caching services (to minimize system response time), and access to a virtual marketplace (e.g., instrument books and market resources) supplied via data center 110 and the core components 112. Together the application server 151 and 152, firewall 153, and trader desks 154 form a trader site 150. The trader site 150 is connected through an external services network 114 to a data center 110. As will be appreciated by those skilled in the art, a network can be maintained on secure, dedicated lines (e.g., a private network) or could incorporate a network such as the Internet. As will be appreciated by those skilled in the art, a trader site 150 could operate with only one server, but may employ multiple servers (as illustrated) depending on volume and resiliency requirements.

Data center 110 includes at least one server 110a. As will be appreciated by those skilled in the art, a "server" is a computer that facilitates the transmission and reception of data between different points such as between a network and external sources. From a hardware standpoint, a server will typically include one or more components for performing the arithmetic and/or logical operations required for program execution, such as one or more microprocessors. A server will also typically include disk storage media such as one or more disk drives for program and data storage and random access memory (RAM) for temporary data and program instruction storage. From a software standpoint, a server computer also contains server software resident on the disk storage media, which when executed directs the server computer in performing its data transmissions and reception functions. Servers are offered by a variety of hardware vendors including Sun Microsystems, Inc. These servers can contain different types of server software, each type devoted to a different function, such as handling data from a particular source or transforming data from one format into another.

In Figure 1, reference numbers 110 and 120 designate data centers. A data center is a trading "hub" for the system 100, through which trading activity and system management within the system 100 are controlled.

In a preferred embodiment, each server in data centers 110 and 120 is a Sun Enterprise 10000 server ("E10k"), manufactured by Sun Microsystems, Inc., or a similar server platform. Documentation for the E10K server can by obtained from Sun Microsystems, Inc. and through Sun's website at http://docs.sun.com (and various links). The server software runs on an operating system such as the Solaris 2.6 platform, developed by Sun Microsystems, which is stored on a disk storage device.

The E10k servers support partitioning. Partitioning is the ability to take a single server and dynamically divide it into multiple smaller systems (or partitions), with each partition running its own instance of the operating system. Each partition is referred to as a dynamic system "domain." Dynamic system domains are logically isolated from each other within the server, which isolated configuration provides a highly secure and reliable environment for running multiple functions simultaneously. Domains can be dynamically configured without interrupting users or production. For example, domains can be created, allocated, and resized while a server is operational. Domain management is achieved through a management console connected to the server.

An example of a basic unit of a domain includes one system board, having up to four processors (such as a "Ultra-SPARC" processor), a four gigabyte memory and four I/O connections. In the example, each domain operates its own instance of the Solaris 2.6 operating environment software, as well as maintaining its own unique name, boot disk, disk connections, memory, CPUs, network and interconnect access--creating a fully functional, fully isolated partition.

Because of the software and hardware independence, most software and nearly all hardware errors in a domain are confined to that domain and will not affect the rest of the server. The E10k server can currently accommodate up to eight domains.

As shown in Figure 2a, the server 110a is preferably configured or partitioned into three domains. The three domains include a database 111, core components 112, and a gateway 113. In an alternative embodiment (e.g., the Figure 2b embodiment), the data centers 110 and 120 each include two E10ks servers, which are each divided into three domains. Including both data centers 110 and 120, twelve (12) domains are provided in the Figure 2b embodiment. As will be appreciated by those skilled in the art, each domain is equivalent to a separate stand alone computer.

Software is installed on broker terminals that produces a broker desk tools (e.g., B-Desk tools) Graphic User Interface ("GUI"), an example of which is shown in Figure 41. As will be appreciated by those skilled in the art, the B-Desk tools software can be stored on computer readable media, including CD-ROMs, floppy disks, removable media, magneto-optical media, fixed disks, flash memory, memory sticks, etc. In a preferred embodiment, brokers support the system and traders through a broker site, like trading site 150 shown in Fig. 1.

System 100 preferably includes an interface to a client database 135. The client database 135 houses information regarding clients and their accounts and various trades made between respective clients. System 100 preferably also includes an interface to a terms and conditions database 136, which houses information regarding fixed income security information. As shown in Figure 1, data centers 110 and 120 are networked to databases 135 and 136. Each data center 110 and 120 preferably includes an interface to communicate with the other data center.

The trading site 150 and data center 110 are also connected through the network 114 to a "backup" gateway 130. The backup gateway 130 connects traders to data center 120, in the event that the local data center 110 is unavailable for service, as discussed below. Gateway 130 could also include a client testing environment (e.g., UAT). Data center 110 is interfaced with external clearing entities 137.

Data center 120 also include a database 121, core components 122, and gateway 123. Data center 120 is preferably located in a different geographical area from that of data center 110.

Database 120 is networked to a trader site 160, including application servers 161 and 162, firewall 163 and user terminals 164, like those described above with respect to trader site 150. The trader site 160 and data center 120 are also connected to a backup gateway 140, as described above with respect to backup gateway 130. The backup gateway 140 could also include a backup CMP. The CMP is described in further detail below.

Figure 1 also illustrates remote client sites 170 and 180. For example, data center 120 may be located in London, while remote trader sites are located in Paris and Frankfurt, respectively. A remote gateway 171 and remote gateway 181 are preferably associated with each remote trading site 170 and 180, respectively. The gateways 171 and 181 serve as an interface between the remote trading sites 170 and 180 and the data center 120. In this regard, the gateway 171 and 181 can each service multiple trading sites, thereby reducing overall connection costs. Without such a gateway, each trading site would need a direct connection to the database 120, which would increase the costs of the overall system 100. As previously mentioned, gateways provide connection services between the components located in the data centers and the application servers. The gateways also provide data caching and filtering to ensure that the data sent to a client or trader site, via his respective application server, is specific to that particular client or site.

The data center 120 also interfaces with clearing entities 138 that are authorized to operate in the corresponding geographic area. Authorization typically requires governmental approval and/or an operating license. In one preferred embodiment, data centers 110 and 120 each interface with clearing entity 137 and 138.

Data centers 110 and 120 preferably are located in distinct geographic locations, with each location having a unique trading market. For example, data center 110 could be located in the United States, with the corresponding trading market focusing on U.S. Treasury instruments. Similarly, Data center 120 could be located in the United Kingdom. By fully integrating the system 100 and data centers 110 and 120, however, a trader in the geographic location of database 110 can trade in both the local market, and in the market associated with the geographic location of database 120, and vice-versa.

As will be appreciated by those skilled in the art, the Fixed Income Trading System 100 is based on an object-oriented design. Objects are defined by sets of data that describe conceptual real world entities and the programming logic that manipulates the data. For example, an "order" object includes all of the data needed to describe an order at a point in time, and the programming functions necessary to process the data. The status of an object reflects the current object's data view. Preferably all processing by system 100 is transactional based. A transaction can be atomic, consistent, isolated or durable. An "atomic" transaction is a single logical unit of work. In an atomic transaction, all changes to an object by a system component are committed to the database 111, as well as the component's cache and an AQ ("Advanced Queuing") synchronization stream. An atomic transaction cannot be partially committed or partially aborted. A "consistent" transaction is a unit of work, which takes an object from one consistent state to another. An "isolated" transaction is one whose partial results during execution are hidden from other objects interacting with the system. A "durable" transaction is one whose results are persistent; that is to say, stored in a database. A transactional system is designed to ensure that all object state changes are recorded as transactions, as discussed above.

In the Fixed Income Trading System 100, objects and associated states are held in the system's databases 111 and 121, shown in Figure 2b. The databases 111 and 121 act as permanent persistent data repositories. In this sense, the databases 111 and 121 track the current snapshot of all system activity. From the view point of data center 120, each component of the core components 112 has an associated cache 118, shown in Figure 18c. The cache 118 acts a virtual data repository for each component in the data center's 110 and 120 servers. The cache 118 provides components with their own "working" copy of the database 111 (or database 121). The distributed working copy improves performance by reducing the database load. As will be appreciated, processing data is faster by manipulating objects stored in a memory cache, as opposed to manipulating data stored on a fixed disk, for example. The cache 118 is preferably a separate cache for each component, however, most of the illustrated drawings refer to a component's cache as reference number 118. In a preferred embodiment, the cache 118 is managed by Power-Tier software, developed by Persistence, Inc.

A technique called "optimistic versioning" is preferably implemented by the cache and a particular target object to guarantee transactional integrity of the system 100. This process is described with reference to Figures 18a through 18c. As seen in Figure 18a, a exemplary component 112x processes data that changes an object's state. For example, component 112x could be a Book Manager 112c, AQ Publisher 112a, or Bulk Request Manager 112g, or the like. In keeping with transaction based processing, the component records the state change by performing a single "commit operation" that is atomic, consistent, isolated, and/or durable. The commit operation is shown with reference to Figure 18b. As shown in Figure 18c, a commit operation simultaneously updates the component's local cache 118, updates the database 111, and creates and broadcasts a synchronization data stream to all system components. The data stream fully reflects the current state of the object. This synchronization data stream is consumed by a cache 118 of each system component to ensure that the component has a current state data image. As a result, the above-mentioned process ensures that all object's state changes result from transaction based processing and the updates reflecting the changes occur simultaneously systemwide. Hence, systemwide object integrity can be ensured. In a preferred embodiment, Oracle 8 Parallel Server, from Oracle, Inc., and Power-Tier, from Persistence, Inc., are used to facilitate simultaneous updating of the database 111 and the caches 118, respectively.

Data Domains

Each of the three domains 111, 112, and 113 will now be discussed in further detail with reference to Figures 1, 2a and 3. While the following discussion is from the perspective of data center 110, it will be appreciated that data center 120 includes corresponding components. Database 111 is a store house for information pertaining to system activity, including order creation, market activity, book status, client information, authorizations and permissions, and final disposition of trades. In a preferred embodiment, the fixed income trading system 100 utilizes Oracle 8 parallel database server software, from Oracle, Inc., for data management and data synchronization, including assisting with a data synchronization data stream. The database components run in their own domain on each server located at data center 110.

The server 110a, for example, maintains two identical images of the system's activity at any given time. A first image is committed to the database 111, and the second image is committed to each component's cache 118. As will be appreciated by those skilled in the art, the image of activity stored in the database 111 is persistent because it is always available, even if the system fails. However, the image stored in a component's cache 118 does not persist if that component fails. The databases 111 and 121 (including software discussed above) also provides the system 100 with a synchronization data stream. Referred to Advanced Queuing ("AQ"), the synchronization data stream provides the system with a data stream that is identical to that which is committed to the database 111. The synchronization data stream is broadcast systemwide by the AQ Publisher 112a, discussed below. The data stream is consumed by system components to update their cache images. Updating each component ensures object state integrity systemwide. The data stream is identical to the information stored in the database 111 because it is created in the same operation that commits the data to the database 111, as well as to the cache 118 of the component generating a change.

<u>Core Components</u>

The system's 100 core components will now be described in further detail. Preferably, a core component is a dedicated task module, or a block of computer executable code (e.g., a software program), that runs on a domain of a server. Alternatively, as will be appreciated by those skilled in the art, a core component could include a combination of software, memory, and hardware. Each component generally operates or controls different aspects of the overall system. Furthermore, components may receive input data from other components, interact with the gateway 113 and database 111, and direct activities throughout the system 100. Each of the system's 100 core components 112 are discussed below, with reference to Figure 3. The following discussion focuses on the components located on the severs 110a and 110b in data center 110. However, it will be understood that corresponding components reside in the servers located in data center 120.

AQ Publisher

The AQ Publisher ("AQP") 112a, shown in Figure 3, is a core component that runs in the active core domain of the servers located in each of the data centers 110 and 120. The AQP 112a facilitates delivery of the synchronization data stream to system components. The synchronization stream is created in the same operation that commits data to the database 111, as well as to the cache 118 of the component generating the change to a given object. Preferably, synchronization data message distribution meets three requirements. First, data messages should be distributed to system components as close to instantaneously as possible. Second, all object state changes (e.g., data updates) should be guaranteed to be available to every component that requires the update. Third, data updates should be provided to components in the order in which the changes occur. Preferably, the AQP 112a facilitates each of these objectives as described below.

To better understand AQP processing, multicast messaging distribution and TIBCO, Inc.'s Rendezvous Distribution is discussed. Multicast messaging distribution is utilized for message delivery throughout the system 100. As will be appreciated by those skilled in the art, multicast message distribution has the following characteristics: (i) a data string consists of multiple logical message queues; (ii) messages are sent in a single broadcast, without establishing verified connections to the applications that need to consume the message; (iii) an assumption is made that applications interested in the messages will be "listening" and will therefore receive the messages; and (iv) multicast distribution is very efficient. A single message broadcast, for example, can service many applications or components.

TIBCO, Inc.'s Rendezvous ("RV") messaging middleware is preferably utilized by the AQP 112a to facilitate multicast message distribution. As will be appreciated by those skilled in the art, the RV middleware converts the data stream into messages in RV format. Messages in RV format employ a subject-based naming convention, with each logical message queue having a unique base subject name. As will be appreciated by those skilled in the art, this type of subject-based naming convention allows the system's components to listen for message streams having a specific name, thereby facilitating the component's ability to listen only for the messages of interest. These messages are used to update each component's cache 118. By filtering data based on subject names, each component's cache 118 contains only the object state information of interest to a given component.

To facilitate multicast distribution, the AQP 112a converts the AQ message streams to RV message streams, and broadcasts the RV messages to other components via the core RV ring. The core RV ring refers to a core network. The RV ring facilitates messaging between applications/components utilizing Rendezvous messaging. The core RV ring including processes called "rv daemons", which are part of TIBCO's Rendezvous product. As will be appreciated by those skilled in the art, a daemon is a software program, which in this case, handles processing for listening for objects with subject-based names.

An example of this distribution scheme is given with reference to Figure 4. An exemplary core component 112x processes data and changes an object's state. By executing a commit operation, the process stream is stored in the component's 112x cache 118 and in the database 111. An AQ message stream is generated, as discussed above. The message stream contains the same information that is contained in the database 111 and in the component's 112X cache 118. The data stream is directed through the AQP component 112a, through which RV formatted message streams for core components are issued.

RV format provides a fast and efficient means for distributing messages, but does not address whether messages are received by individual components. To determine whether a message is received, a sequence number is assigned to each message within a given data stream. The core components 112 monitor the sequence numbers as messages are consumed from a given data stream. If the progression of sequence numbers is interrupted, the component recognizes that a message represented by a missing sequence number has been missed. If sequence messages are missed, the component in question requests a retransmission of the missing messages from the AQP 112a. The AQP 112a sifts back through the AQ stream to find the missing message and retransmits the missing message to the complaining component.

The system 100 provides a procedure for core components to recognize an AQP 112a failure. The AQP 112a periodically sends "heartbeat" messages at regular intervals to components in the event that an AQ synchronization stream does not contain any new status change messages for a given component. A component may not receive AQ messages for several different reasons. The first reason is simply part of normal system operations, for example, when there are no messages in the synchronization string that apply to the given component.

Hence, the component will wait for a relevant message to be issued. The second reason that a component may not receive messages indicates a potentially serious problem, e.g., that the AQP 112a has failed, and is not sending messages. The various core components 112 realize that the AQP 112a is operational as it receives either heartbeat messages or sequenced synchronization message traffic. If the components 112 do not receive either of these messages, they may indicate to the system 100 that a potential problem with the AQP 112a exists.

An example of the function of the AQP 112a will be described with reference to Figure 5. The AQP 112a receives an AQ message from the database 111. The AQP 112a processes the message by converting it into RV format. As seen in Figure 5, a consuming component 112x receives data messages 1, 2, and 4. As will be appreciated by those skilled in the art, component 112x realizes that an interruption in the data stream has occurred. The consuming component 112x issues or transmits a message 3 missing signal to the AQP 112a. In response, the AQP 112a sifts back through the data stream to find and reissue the message number 3 to the core RV ring. The missing message 3 is transmitted through the core RV ring and is consumed by component 112x.

The operation of the AQP 112a is further described with reference to Figure 6. The AQP 112a transmits four messages as in the last example and they are each successfully received by the consuming component 112x. In the example, the four messages are followed by a period of time where no synchronization messages are sent that are applicable to the consuming component. Hence, the component 112x does not receive any messages. The AQP 112a sends "heartbeat" messages so that the component knows that the AQP 112a is still operational. In a preferred embodiment the heartbeat message has the same sequence number as the last synchronization message.

DR Replicator

The DR Replicator ("DRR") 112b, shown in Figure 3, is a core component that runs in the active core domain on a server in each of data centers 110 and 120. The DRR 112b ensures that a duplicate image of database 121 resides in data center 110, and a duplicate image of database 111 resides in the data center 120. Preferably, database replication occurs in real-time.

Databases 111 and 121 are repositories for information pertaining to all system activity, including books, trades, orders, order creation and order disposition (e.g., cancel, expiration and/or execution). Data bases 111 and 121 could also contain information regarding clients, trade history, and market activity, for example. Preferably, database 111 contains information specific to a market located in an area associated with data center 110. Likewise, database 121 preferably contains information specific to a market located in an area associated with data center 120.

Real-time replication ensures that a current replica of each database is maintained in a geographically separate location in the event that data center 110 or 120 is not available for service. By maintaining a geographically separate copy of each marketplace's database, system recovery is ensured in the event of a failure, e.g., the original database is destroyed or is otherwise unavailable. One purpose of maintaining duplicate databases is to ensure that records of all trades made up to the failure are safely recorded so clearing and settlement can be assured. In a failure scenario, the surviving server would be used to process the duplicate database to facilitate clearing and settlement or to resume trading activity for the alternate market. In the Figure 2b arrangement, databases 111 and 121 within each data center 110 and 120, respectively are redundant as well. That is to say that each data center 110 or 120 has two identical databases 111--one residing on each server. Within data centers 110 and 120, database replication between connected servers (i.e., between servers 110a and 110b, or between servers 120a and 120b) is preferably facilitated via Oracle, Inc.'s Parallel Server technology, which is part of the Oracle 8 software system. As will be appreciated by those skilled in the art, any other software platform which allows duplicate replication between servers would be acceptable for this process.

Both data centers 110 and 120 preferably have identical copies of the local and the geographic separate data center's database. For example, servers 110a and 110b each have a copy of database 121. Similarly, each of servers 120a and 120b maintain a copy of database 111 and database 121. Typically, only one instance of the DRR 112b runs within a given data center 110 or 120. The DRR 112b receives an AQ message stream generated by the database associated with the server in the other data center. For example, the DRR 112b in the data center 110 receives an AQ stream generated by database 121 in the data center 120. The DRR 112b receives and processes data center's 120 stream so that it can save data center's 120 database 121 within its own data center 110.

An example of offsite replication is shown in Figure 7. As seen in Figure 7, the AQP 112a translates the data stream into an RV message, which is relayed to the core RV ring. The core RV ring provides the messages to component 112x and to the database 111. The database 111 issues an AQ synchronization stream to the AQP 112a and to the DRR 122b of data center 120a. The DRR 122b then stores a copy or image of the market associated with the data center 110a in the database 127. As will be appreciated by those skilled in the art, the database 127 can be physically housed in either the core components 112 domain or in the database 111 domain itself. At this time, the server 110a can also transfer an image or replica copy of the database 111 to the second server 110b for backup storage. Data center 120a also transfers a duplicate of its trading market through the database 121 to the DRR 112b. A real-time image of the market associated with the data center 120a is stored in database 117.

As previously mentioned, a single instance of the DRR 112b runs in each data center 110 or 120. Should this instance fail, recovery can be facilitated by manual intervention or recovery software that restarts the failed instance or starts a new instance. During the period that the DRR 112b is out of service, the database 111 (or an AQ generating component resident in the database 111) builds a queue to store all messages that are to be consumed by the failed replicator. Software provided by Oracle, Inc., preferably Oracle AQ, can assist in delivering the queued messages once a new instance of the DRR 112B is started, for example. The DRR 112b is started and configured by the master control process, discussed below, when the system is started.

Master Control Process

The master control process ("MCP") 119, shown in Figure 8, is a component that runs in each gateway domain and core component domain of each server located in data centers 110 and 120. In the Figure 2b embodiment, two instances of the MCP run on each server. The MCP 119 is responsible for starting and configuring the components running in a given MCP's 119 server domain. For example, with reference to Figure 3, the MCP 119 in the core component's domain 112 is responsible for starting and configuring each component in that domain. Similarly, the MCP 119 in the gateway 113 domain is responsible for starting and configuring that particular gateway 113. The MCP 119 also provides system administration functionality, including monitoring the core and gateway components status as well as facilitating user command options. The MCP 119 also supports the system's 100 resiliency functions. The MCP 119 component has four main functional units, including a MCP unit, a mcp config manager, a mcp_cfg editor, and a MCP monitor.

The MCP unit is the processing center of the MCP 119 component. The MCP unit manages and monitors the components running in its particular domain. The MCP config manager supplies all of the MCPs 119 within a data center with a centralized configuration information repository. The MCP config manager enables administration to be more efficient, since all MCPs 119 can utilize the single source of configuration information. Without this arrangement, each MCP 119 would require a local configuration file, which would have to be administered. As seen in Figure 8, the MCP config manager can be remotely located (e.g., the MCP config manager does not have to be on a server or in a data center 110 or 120).

The mcp_cfg editor provides an administrative interface to edit the configuration information supplied by the MCP config manager. Like the MCP config manager, the mcp_cfg editor can be remotely located, as shown in Figure 8. The MCP monitor provides a monitoring and command interface to the MCPs 119 for administrative purposes. The MCP monitor can also be remotely located. The MCP monitor is the primary interface for monitoring and controlling components that run in a given MCP's 119 domain. The MCP monitor provides status information, window command features (e.g., point and click to execute a commands), as well as commands for issuing text space. The MCP monitor can display the processing status for each component in all MCP managed server domains. The MCP monitor is used to manually control the components running in a given domain. As shown in Figure 8, each of the MCP modules are connected via an RV ring.

In the Figure 2b embodiment, the core and gateway domains necessary to run the system 100 can reside on one server or they can be split across both servers in a particular data center 110 or 120. In Figure 9, the "active" core domain runs on server 110b and the "active" database and gateway domains run on server 110a for example. Splitting the domains across two servers may be an efficient way to share resources and to ensure efficient redundancy between linked servers, for example. The MCP 119 designates which domains are to be used as active domains in start-up scripts executed after a domain's operating system boots. If a domain is to be utilized to run the system, the script will start an MCP 119 instance, which will in turn start and configure the components within its domain.

The MCP 119 configures a domain based on system information, including database connection information, RV connectivity information, identification of shared libraries, the Persistence model being used, how many instances of a given component should be started (e.g., for multi-instance components like the Book Manager 112c), etc. A Persistence model can relate to the Power-Tier product and/or the system's 100 memory object relational database, for example.

A system startup and configuration, and the MCP's 119 role in the startup process, is now described. First, servers 110a, 110b, 120a and 120b are powered up and the operating system in each domain boots. Startup scripts are then executed and the

MCP 119 instance in each domain is started. The startup scripts determine which MCPs 119 should start their components, and which should not. MCPs 119 that do not start their components remain operational in order to be available to start their components in the event the corresponding "active" domain fails. These MCPs 119 that are instructed to start their components broadcast a configuration request message via RV. This message is consumed by the mcp config manager. If an mcp config manager is running, the mcp config manager provides the MCP 119 with all configuration instructions necessary to start the components for which it is responsible. If MCPs 119 do not find an mcp config manager, the MCPs 119 can use a local copy of the configuration information used when the system was last started. Each MCP 119 automatically creates this file when it starts the components in its domain. The MCP 119 polls each component to determine its status. This "status" information is then broadcast via RV messaging to listening MCP Monitors.

As previously discussed the MCP 119 plays a central role in system resiliency functionality. Before discussing the MCP's 119 role in system resiliency, however, the following points are noted. First, the domains necessary to run the system 100 can reside on one server or they can be split across both servers in a data center, as previously discussed. From a resiliency perspective, this is important because the system preferably does not utilize a traditional "hot" or "cold" standby model where a host computer is actively processing while a backup host computer waits to go live in the event that the active host fails. In the Figure 2b embodiment, each active domain in the system has a backup domain on the other server in a given data center that is available to take over processing should a problem arise in the operation of the components in an active domain. Second, if a component fails the MCP 119 preferably automatically attempts to restart it.

However, if the component can not be restarted, manual intervention may be necessary to solve the problem, either by fixing the failed component so it will restart or by starting the backup domain. Third, automatic domain-level failover is supported. For example, if a domain fails (e.g., a hardware or operating system failure causes a failure), the MCP 119 running in the backup domain on the other server will automatically start the components necessary to replace those in the failed domain.

If the MCP 119 detects that a component has failed, it automatically attempts to restart the component (e.g., makes five attempts over a three minute period). As will be appreciated, the MCP 119 can be configured to make any number of attempts over any given time period.

Domain failure can occur due to a hardware or operating system problem. The MCP 119 facilitates automatic failover in the backup domain of the adjacent server in the event that the active domain fails. For example, referring to Figure 9, if the core components of server 110b failed, the MCP 119 automatically converts failover to the core component domain of server 110a. The MCP's 119 role in system failure is further described with reference to Figure 10. In Figure 10, server 110a houses the active database 111 and gateway 113, while server 110b houses the active core components 112. As shown an MCP 119 instance resides in all core component 112 and gateway 111 domains. Each MCP 119 instance exchanges "heartbeats" with its counterpart MCP 119 via an RV line. For example, as seen in Figure 10, the MCP 119 resident in the gateway 111 domain of server 110a exchanges a heartbeat with a counterpart MCP 119 resident in the gateway domain of server 110a.

In a preferred embodiment, cluster management software provides the system 100 with a procedure for system level status checking between the domains on each of the servers 110a and 110b (or between servers 120a and 120b). Such cluster management software is developed by Sun Microsystem, Inc.'s, for example, including Sun's HA cluster management software. The HA technology provides a procedure to determine whether a problem is related to a domain failure. In one example, the core components remain operational, but a RV ring interrupts the MCP 119 heartbeat exchange. Without cluster management software, the backup MCP 119 could erroneously conclude that the other MCP domain had failed, since it is no longer receiving heartbeats from that domain. If the HA indicates a system level problem, the MCP realizes that the problem is related to a network or application issue. The HA technology utilizes a private data channel for checking system status. Preferably, the data channel is a separate channel.

If the HA software relays a problem signal, the backup MCP automatically starts all of the components in the backup domain. In this manner the MCP 119 enables the system to resume processing without data loss in the event of a domain or server failure. In the event that a backup MCP fails to receive heartbeats from an active MCP, but does not detect a problem through the HA cluster management software, the backup MCP waits for a predetermined amount of time (e.g., five seconds or so) to see if the heartbeats resume, as would be the case if a network issue caused a momentary interruption. If the heartbeats are still undetected, MCP 119 logs messages to indicate that manual intervention may be necessary to resolve the problem, as the problem could be caused by a network or application issue.

The MCP 119 has various interfaces as discussed above. For example, the heartbeat messages are broadcast via RV channels between MCP 119 instances in corresponding domains. Configuration information is broadcast via the RV channel to all components running in the same domain. Also, status messages are broadcast via RV to the MCP monitor.

The MCP 119 also evaluates various inputs, for example, heartbeat messages received from the MCP 119 instances in a corresponding domain. The MCP 119 also receives as an input, configuration information received from the MCP config manager. Also, systems status information is received from the cluster management software, as previously discussed.

Each MCP 119 is started by a boot script that executes when the domain's operating system boots. Manual control of a MCP 119 is executed by using the MCP monitor, as previously discussed. The MCP 119 continuously runs under normal conditions and is not stopped unless the system is completely shut down.

Book Manager

The Book Manager 112c, shown in Figure 3, is a core component that runs in the active core domain of a server located in the data centers 110 and 120. Preferably, each book on the system 100 has a corresponding Book Manager 112c instance. As discussed, a book in the system 100 can be defined as a set of bids and offers that compete with each other for the same instrument and settlement date in the same market type. If bids or offers exist for multiple settlement dates for a single security, they are preferably maintained in separate books.

The Book Manager 112c is responsible for managing the system's 100 order books. In particular, for passive orders, the Book Manager 112c adds the order to a given book, after the order request is validated. A passive order is a firm bid/offer submitted by a trader with the intention of being displayed in a book for a particular instrument. Validating a request preferably includes verifying that the requested instrument is found in database 111 (e.g., a book exists) and the requested instrument is tradeable (e.g., the instrument has not expired). The Book Manager 112c also ensures orders are sorted and/or listed appropriately within the book. Additional features of the Book Manager 112c include processing user requests to modify, suspend, or cancel passive orders. The Book Manager 112c also "expires" orders based on requests issued by the Expiry Manager 112e, discussed below. The Book Manager 112c can also be used to open and close a book according to the direction of the Rollover Manager 112f, discussed below. The Book Manager 112c also responds to suspend requests by the Bulk Request Manager 112g, discussed below.

The system 100 provides several different subsets of information regarding each book (e.g., different book views). As will be appreciated by those skilled in the art, the subsets could be created and maintained by the Book Manager 112c, or a book could be filtered (e.g., by the GUI, discussed below) to obtain desired subsets of information. A "Depth of Book" view for an instrument displays real-time information about its active orders. Examples of information displayed in the Depth of Book include: i) an ordered list of price/time sequenced bids and offers; ii) most recent trade information such as amount, price, time, hit/take, and indicator; and iii) settlement date information. Bids are preferably sequenced from highest to lowest price. Offers are preferably sequenced from lowest to highest (for currency price) or sequenced from highest to lowest (for discount and yield orders). Orders are preferably sequenced from oldest to newest within a single price.

A "Top of Book" view refers to information about the best bids and offers for a given security on the fixed income trading system. This view may also be referred to as the "market level," since it gives a view of instruments and their associated top bids/offers level. The "Key Issues" view provides information which allows a user to create a customized view of the top of the book activity for a specific set of securities.

A status of a book describes the book's ability to accept orders. A book's status can be changed automatically by the system or by a broker using the B-Desk tools. Book statuses include: 1) open, which indicates that a book can accept orders for a security that it represents; 2) closed, which indicates that a book can not accept orders; 3) inactive, which indicates that a book is in maintenance mode; and 4) retired, which indicates that a settlement date for a book is past maturity.

The system automatically assigns the book an "inactive" status during rollover. Rollover is a term used to describe the process of the system 100 calculating a new settlement date for the security. In a preferred embodiment, a broker (e.g., a party or entity that is neutral with respect to the outcome of a transaction) can change a book's status to "inactive" using various B-Desk tools, as discussed below. Changing a status to inactive is performed to prevent trading in a given book (e.g., to limit the security's availability for trading). As discussed, users or traders can submit orders such as a passive order with the intention of the order being displayed in the book.

For aggressive orders, the Book Manager 112c effects trade executions. An aggressive order is an attempt to hit or lift a passive order. A successfully aggressed order results in an executed trade. Alternatively, an execution could include a "failed" order attempt and/or a failure notification. Assuming full execution, an executed passive order is removed from its respective book. An aggressive order is transient, meaning that it is either filled immediately against one or more passive orders, posts a residual amount as a passive order, or it fails. Executions are used by the Trade Manager 112d to create real trades or ledger transfers, as discussed below.

A book keeps an order record for a given security and an associated settlement date. Each book on the system is managed by a single instance of Book Manager 112c. Therefore, there are multiple instances of Book Manager 112c in a given domain on a server, each managing a single book. Before the Book Manager 112c can administer a book, the book must exist on the system. The term "exist" means that objects have been created for the book, and that the system is aware of the book. However, the term "exist" does not necessarily imply that the book is available for order activity. For example, the status of a book could be closed, in which case the book is not available for order activity. To be available for orders, the book must have an "open" status. Open/closed status is controlled by brokers using the B-Desk tools, as discussed below. Orders other than "good til canceled" expire on a book closure (which is done using the B-Desk tools or the Rollover Manager 112f doing rollover) and requests for new orders are rejected until the relevant book is reopened.

The Book Manager 112c interacts with several other core components. The Expiry Manager 112e consumes executions made by the Book Manager 112c. The Expiry Manager 112e consumes order messages to determine whether it must track expiree criteria. Gateways 113 and 123 consume all order output in order to pass relevant information to appropriate application servers. "Appropriate" in this instance could mean that the information is intended for a given application server's client, for example.

The Expiry Manager 112e sends expiration requests to the Book Manager 112c, based on its processing of orders having expiry criteria. The Rollover Manager 112f sends cancellation requests to the Book Manager 112e when the Rollover Manager 112f sets a book status to an inactive status during rollover. The Bulk Request Manager 112g sends suspended requests to the Book Manager 112c when the Bulk Request Manager 112g suspends all commands due to a system or user input.

The Book Manager 112c is started and configured by the MCP 119 automatically when the system is started. Manual control of the Book Manager 112c is obtained using the MCP monitor, as discussed above. The Book Manager 112c preferably runs continuously, unless it is stopped to facilitate system maintenance. The various Book Manager 112c instances running on the servers respond to book status changes executed by the Rollover Manager 112f (e.g., going from an "open" status to an "inactive" status). The Book Manager 112c processes order requests such that it will reject order requests while the status of a book is inactive. However, an instance of the Book Manager's 112c remains active on the server even though the book's status is inactive.

Trade Manager

The Trade Manager 112d, shown in Figure 3, is a core component that runs in the active core component domain of the data centers 110 and 120. The Trade Manager 112d is responsible for processing executions made by the Book Manager 112c. Specifically, the Trade Manager 112d creates either "real trades" or "ledger transfers." A real trade is one that will settle. A ledger transfer is a situation where a trade is executed, but the buyer and seller are from the same firm. A ledger transfer will not be settled, but the Trade Manager 112d handles the necessary reporting to account for such a trade record. Preferably, with a ledger transfer, the Trade Manager 112d generates a notification to be relayed to the buyer and seller informing them that they are from the same firm. In the preferred embodiment, a commission is not charge for a ledger transfer. The Trade Manager 112d also performs post trade calculations, such as coupons, accrued interest, and fee calculations, etc.

The Trade Manager 112d creates "tickets". A ticket is a record regarding trade details, and includes information about settlement, price, amount, instrument, etc. A client can manually print a ticket through the trader graphic user interface. A trader can use information from the ticket to be entered into his own trade capture system, for example. For the U.S. market, for example, a single ticket cannot exceed more than $50 million dollars. The $50 million dollar limit is the Federal Wire limit on fund transfers (all U.S. government fixed income transactions settle via the Federal Wire). When processing a U.S. trade, the Trade Manager 112d breaks the trades into blocks as necessary to ensure the Federal Wire requirements are met. The Trade Manager 112d also produces a "ticket set," which is a message that associates all of the individual tickets with the single execution that results in the ticket generation. In a "sweep the book" execution involving n orders, as discussed below, the Trade Manager generates n+1 tickets (e.g., 1 ticket for the aggressing order, and n tickets corresponding to the n "swept" orders). For a European market, the Trade Manager 112d is configured so as not to break-up executed trades, since there is currently no Federal Wire-like requirement in Europe.

The Trade Manager 112d interacts with several other core components. For example, the Back Office Adapter 112k, shown in Figure 3 consumes the tickets and ticket sets output from the Trade Manager 112d for settlement processing. Also, the Book Manager's 112c messages regarding executions are consumed by the Trade Manager 112d.

The Trade Manager 112d is started and configured by the MCP 119 automatically when the system is started. As with the other core components, manual control of the Trade Manager 112d is executed using the MCP monitor. Under normal conditions the Trade Manager 112d runs continuously except when it is stopped to facilitate system maintenance.

Expiry Manager

The Expiry Manager 112e, shown in Figure 3, is a core component that runs in the active core domain of a server in both of the data centers 110 or 120. The Expiry Manager 112e is responsible for ensuring orders expire according to criteria submitted by the user. In particular, the Expiry Manager 112e captures order traffic being generated by the Book Manager 112c to determine whether an order submission specifies expiration criteria. For orders with expiration criteria, the Expiry Manager 112e calculates the expiration time, and sets a timer to signal when the expiration time occurs. The Expiry Manager 112e sends an expiration request message that is consumed and processed by the Book Manager 112c when an order expires. Preferably, the Book Manager 112c then removes the order from the book.

The Expiry Manager 112e interacts with various other core components. For example, the Expiry Manager's 112e "expire" messages are received and processed by the Book Manager 112c, as discussed above. Order messages from the Book Manager 112c are received and processed by the Expiry Manager 112e to determine whether expiration criteria must be tracked for a given order. Like the other core components the Expiry Manager 112e is started and configured by the MCP 119 automatically when the system is started. Furthermore, manual control of the Expiry Manager 112e is accomplished using the MCP monitor. Except for system maintenance, the Expiry Manager 112e continuously operates.

Rollover Manager

The Rollover Manager 112f, shown in Figure 3, is a core component that runs in the active core domain of a server at the data centers 110 and 120. The Rollover Manager 112f ensures that rollover occurs for all instruments available for trading in the system. "Rollover" can be defined as the point at which a settlement date for a given security expires and a new settlement date is calculated. For example, if an instrument had a settlement date of T + 1 ("T" representing a trade date), there is a point in the trading day at which it is no longer possible to execute a trade and settle according to the T + 1 requirement. The Rollover Manager 112f ensures that no trades occur after this point, and that Rollover Manager 112f resets the settlement date of the instrument to the next applicable date available.

In particular, the Rollover Manager 112f calculates the time at which trades for a given security can no longer be made. For this process, the Rollover Manager 112f changes the status of the book for the instrument to "inactive." The status change signals the Book Manager 112c to cancel any outstanding orders in the book, if the book had been in open status. The Rollover Manager 112f then calculates the next settlement date for the security as discussed above. The Rollover Manager 112f then returns the book to its previous status.

The Rollover Manager 112f manages the status of each book. Status refers to a book's ability to accept orders. As discussed above, a book has at least three statuses: (i) open (i.e., accepting orders); (ii) closed (i.e., not accepting orders); and (iii) inactive (a status that is similar to closed in that the book is not accepting orders). "Inactive" is the status a book is given when it is associated with a security that is going through the rollover process. The Rollover Manager 112f changes a book's status according to requests made by brokers associated with the system 100. These brokers use the B-Desk tools as discussed below to manipulate the statuses of the book. For example, at the start of each day, books must be "opened," or their statuses must be changed from "closed" to "open."

The Rollover Manager 112f is responsible for receiving requests from the B-Desk tools, validating the request, and changing a book's status as needed. When validating a request made by an affiliated broker to create a new book, the Rollover Manager 112f ensures that the requested instrument is available in the system's database 111, and that the security is authorized to trade on the system 100, for example.

The Rollover Manager 112f interacts with the Book Manager 112c regarding a book status change. The Rollover Manager 112f verifies requests from the B-Desk tools during the process of book creation, for example. The Rollover Manager 112f is started and configured automatically by the MCP 119, shown in Figure 8, when the system is started. Manual control of the Rollover Manager 112f is accomplished using the MCP monitor. Except for system maintenance, the Rollover Manager 112f operates continuously under normal conditions.

Bulk Request Manager

The Bulk Request Manager 112g, shown in Figure 3, is a core component that runs in the active core domain of a server at data centers 110 or 120. The Bulk Request Manager 112g is responsible for processing bulk requests to cancel or suspend a user's order. Such a request can be generated by the system or by the user. For example, a system-issued suspend request occurs when gateway 113 loses contact with a client's application server. The gateway 113 immediately suspends the client's orders to safeguard the client's position until communication is reestablished with the client. In a bulk cancel/suspend scenario, the Bulk Request Manager 112g determines which books contain orders affected by the bulk cancel/suspend request. The Bulk Request Manager 112g then communicates with the Book Manager 112c to cancel/suspend the appropriate orders. As discussed, the Book Manager 112c receives the cancel/suspend request resulting from the Bulk Request Manager 112g processing. The Bulk Request Manager 112g receives as input signals messages from the system 100 or from traders using the system 100.

As with the other core components, the Bulk Request Manager 112g is started and configured by the MCP 119 automatically when the system is started. Manual control of the Bulk Request Manager 112g can be obtained using the MCP monitor. The Bulk Request Manager runs continuously under normal conditions, except when it is stopped to facilitate system maintenance.

Session Manager

The Session Manager 112h, shown in Figure 3, is a core component that runs in the active core domain of a server at data centers 110 and 120. The Session Manager 112h is responsible for managing user requests to access the system. In particular, the Session Manager 112h authenticates trader log in information and allows one trading session per log in.

The Session Manager 112h is started and configured by the MCP 119 automatically when the system is started. Manual control of the Session Manager 112h is accomplished using the MCP monitor. Under normal conditions, the Session Manager 112h runs continuously, except for when it is stopped for system maintenance.

Terms and Conditions Adapter

The Terms and Conditions Adapter 112i, shown in Figure 3, is a core component that runs in an active core domain of a server at data centers 110 and 120. The Terms and Conditions Adapter 112i provides an interface between the system 100 and the Terms and Conditions Database 136 (see Figure 1). The Terms and Conditions Database 136 is the originating source for all fixed income security information for the system 100. Database 136 includes information such as CUSIP, ISIN, description data, terms and conditions, etc. An example of the database may be Reuters' Financial Data Model Fixed Income Product ("RFDMFI"). The components of the Terms and Conditions Adapter 112i and their associated functionality are discussed below. A receiver module receives terms and conditions updates and populates a database 139, preferably located in data center 120. In an alternative arrangement, database 139 resides in either data center 110 or 120, or both. Database 139 is referred to as an "intermediate" database.

An updating process updates the Adapter's 112i cache and the system's core database 111 to reflect updates to the intermediate database 139 (e.g., to reflect terms and conditions updates). Database 111 propagates updates to the core RV ring via the AQ synchronization stream. Alternatively, brokers associated with the system can use the B-Desk tools to select an instrument that should be available for trading from the intermediate database 139, and thereby update the system's core database 111 regarding that instrument, or the terms and conditions regarding the instrument. If the brokers have direct access as discussed above, the updating process could be eliminated.

A variety of other components are utilized to supply instrument information to the system 100 via the Terms and Conditions Adapter 112i. The process as shown in Figures 11a, 11b, 11c, and 11d will now be described. Referring to Figure 11a, database 136 supplies terms and conditions data to the system 100. A dequeuing and converting module is provided to receive the data from the database 136 and to translate the stream into RV messages. For example, a product called "TIBAQ" from TIBCO, Inc. can be used for this purpose. RV messages are transmitted from the TIBAQ to a local RV ring.

A Terms and Conditions Adapter 112i module, called a “tac_receiver,” for example, receives and stores messages in the intermediate database 139, located in the data center 110 or 120. As will be appreciated, duplicate copies of the intermediate database 139 can be maintained on separate servers or in separate locations. For example, a redundant database for data center 120 can be maintained on either the data server 120a or 120b or in the data center 110.

The intermediate database 139 contains the instruments available through the Terms and Conditions Database 136. The system 100 copies the data to the intermediate database 139 to avoid data corruption or compatibility problems with outside applications, for example. In this regard, if an application was designed to use a specific data scheme that was then changed, the application would need to be changed accordingly to accommodate the new scheme. Furthermore, by copying information from the outside database 136, the system can activate any fixed income instrument without having to involve an outside entity. Instruments are made available to the system 100 data centers 110 and 120 by a Terms and Conditions Adapter module, called, for example "tac man." The tac man module listens on a local RV ring for messages pertaining to the market associated with the hosting data center (e.g., U.S. or Europe). When a new instrument message is received, tac man updates the Term and Conditions Adapter's cache 118, the database 111, and the AQ data stream. This is one way that a new instrument can be added to the system 100. Alternatively, the tac man module can be called into action manually by a system 100 administrator (or a broker), who is given the task of adding an instrument to the system. New instruments may be made available to the fixed income trading system by brokers creating books for new instruments via the B-Desk tools, for example.

Figure 11c illustrates how an instrument is added manually to the system. A system 100 administrator 201 selects an instrument from the intermediate database 139 via a sqlplus module which generates a message on a tac RV ring via the TIBAQ 137. Sqlplus is a product from Oracle, which enables manipulation of database commands and database management. The tac man 202a and 202b modules in the data centers 110 and 120 respectively, listen to the tac RV ring for messages pertaining to the market associated with the hosting data center. When a message is received, the tac man 202a and 202b update the Terms and Conditions Adapter's cache 118 and database 111, which in turn updates the core RV ring via the AQ synchronization stream as discussed above.

Figure 11d illustrates an example of a broker 210 directly adding an instrument to the system using the B-Desk tools. In this case, the B-Desk tools directly update the B-Desk application server's cache, which propagates the change to the database 111 or 121, and an AQ synchronization stream is issued. With this configuration, the tac man 202a module is not needed, since the instrument is added via the B-Desk application server within a data center.

As with the other components, the Terms and Conditions Adapter 112i outputs messages to the Rollover Manager 112f. The Back Office Adapter 112k, as discussed below, also consumes messages regarding security information for the Terms and Conditions Adapter 112i. The Terms and Conditions Adapter 112i consumes messages generated by the intermediate database 139 and messages distributed by the core's RV ring. The Terms and Conditions Adapter 112i is started and configured by the MCP 119 automatically when the system is started. Manual control of the Terms and Conditions Adapter 112i is accomplished using the MCP monitor. The Terms and Conditions Adapter 112i runs continuously except when it is stopped to facilitate system maintenance.

Client Database Adapter

The Client Database Adapter 112j, shown in Figure 3, is a core component that runs in the active core domain of a server at data centers 110 and 120. The Client Database Adapter in the data center 110 maintains client information regarding the market associated with the data center 110. Similarly, the client database adapter associated with the data center 120 preferably maintains a client database corresponding to the local market. For example, if the data center 110 was located in New York, the Client Database Adapter preferably maintains U.S. client data. The Client Database Adapter 112i also provides an interface between the system 100 and the Client Database 135. The Client Database 135 stores information regarding client accounts, including client organization, client information, user i.d., permission, organization level, company level, broker fee discount information, broker to client coverage, accounts/privileges, and credit limit settings, for example.

A description regarding how client data is supplied via the Client Database Adapter 112j will now be given with reference to Figures 12a through 12c. The Client Database 135 publishes RV messages to a local RV ring via a client database publisher 220. Like the AQP 112a, discussed above, the client database publisher 220 converts data messages into RV messages. Referring to Figure 12b, each Client Database Adapter 112j and 122j subscribes to messages on the local client database RV ring, which was provided by a cdb publisher 220. The cdb publisher 220 converts information from the client database 135 into RV format. Each client database adapter 112j is preferably only interested in messages pertaining to clients associated with that center's trading market place. For example, the Client Database Adapter 112j is interested in those messages pertaining to the market associated with the data center 110. Messages are received by modules (e.g., called "cdbman") 221a or 221b, which are responsible for processing and receiving messages from a client database 135 local RV ring. When the cbdman 221a or 221b receives a message it queues the Client Database Adapter 112j to simultaneously update its cache 118, database 111, and the AQ synchronization stream in a single commit operation. Preferably, client data is propagated through the system in this manner.

The Trade Manager 112d consumes messages output from the Client Database Ddaptor. Similarly, the Session Manager 112h consumes messages output from the Client Database Adapter 112j. The Client Database Adapter 112j consumes messages generated by the Client Database 135. The Client Database Adapter 112j is started and configured by the MCP 119 automatically when the system is started. Manual control of the cdb adapter is accomplished using the MCP monitor. The Client Database Adapter 112j runs continuously except when it is stopped to facilitate system maintenance.

Back Office Adapter

The Back Office Adapter 112k, shown in Figure 3, is a core component that runs in the active core domain of a server at data centers 110 and 120. The Back Office Adapter 112k listens for tickets or ticket sets from the Trade Manager 112d, then transmits the tickets associated with each set in real-time to an outside clearing entity 137. The Back Office Adapter 112k manages ticket status. A ticket status indicates how a ticket is handled by an outside clearing entity, as discussed below. The Back Office Adapter 112k also logs trades to a database file. The file can be used for manually transferring trade information to the outside entity or clearing house, in the event of a real- time transmission failure.

Figure 13a illustrates a schematic view of the Back Office Adapter 112k, and its relationship to the outside clearing entity 137, MQ manager 193 and monitoring interfaces 194 and 195. The monitoring interfaces can be a terminal emulation 194 of an IBM 5250 for example and/or a software tool called a ticket manager 195, which is available for brokers through a broker desk. The ticket manager tool 195 runs on a workstation which gains access to ticket information through an application server. The application server communicates with the gateway 113, for example. Brokers monitor whether tickets are successfully transferred to the outside entity or clearing entity 137 using the ticket manager tool 195. A TCP/IP link from the MQ manager 193 can be realized using a standard line, for example, a 56K line, DSL line, or T1/T3 line or higher, as will be appreciated by those skilled in the art. Alternatively, the link can be maintained as part of a ESN (e.g., "external services network) 14. The IBM 5250 emulation system can be used to monitor whether tickets are successfully transferred to the outside clearing entity 137.

Figure 13b illustrates an example of the procedure to facilitate transmission of tickets to the outside clearing entity 137. The Trade Manager 112d creates ticket sets associated with tickets executed by the Book Manager 112c, and outputs the ticket sets to the core RV ring in an exportable status. At this time the Trade Manager 112d also simultaneously updates the cache, database 111 and issues an AQ synchronization data stream back to the core RV ring.

The Back Office Adapter 112k processes the ticket sets in the exportable status, and sends the associated tickets to the clearing entity 137 via the MQ manager 193. The Back Office Adapter 112k logs the ticket sets in a file. This file can be used for manual transfer to a clearing entity 137 in the event of a problem or as a quality insurance check, for example. The MQ manager 193 builds a queue of ticket messages and transmits the queue to the clearing entity 137 which preferably uses a MQ interface. Figure 13d illustrates the relationship between the MQ manager 193 and the clearing entity's 137 MQ manager 193.

There are four possible ticket statuses associated with the Back Office Adapter. These statuses include "exported," "accepted," "broken," and "pending." In the event that the MQ manager 193 issues no report, the ticket status is changed to exported. An exported status means that the Back Office Adapter 112k will interpret that the ticket was transmitted successfully to the clearing entity 137. An exported status is not a confirmation from the clearing entity 137 that the ticket was received or accepted for processing. Even though a ticket status is exported, it is still possible that the ticket was not transmitted, or that it was transmitted but not accepted for processing by the clearing entity 137. It is also possible that the clearing entity 137 received and accepted the ticket, but the confirmation message was not received back by the MQ manager 193.

If a ticket is acknowledged by the clearing entity 137 as having been accepted for processing, the ticket status is changed to accepted. An accepted ticket status means that the clearing entity 137 has received and accepted the ticket for processing, and the Back Office Adapter 112k has received this acknowledgment. If the ticket does not export successfully, yet the MQ manager 193 reports that its connection to the clearing entity 137 is operational, the ticket status is changed to broken. A broken ticket status indicates a problem with the ticket or a technical issue preventing the transmission of the ticket. If the ticket does not export successfully and the MQ manager 193 reports that its connection with the clearing entity 137 has been severed, the ticket status is changed to pending. A pending ticket status means that the MQ manager 193 will cease adding tickets to the queue. Tickets will be maintained by the system until the problem is fixed. Any tickets already in the queue when the problem begins will remain there and should be sent when the problem is fixed. Hence, a pending ticket status means that tickets are not being sent from the system to the clearing 137. In this case the MQ manager 193 will attempt to reestablish the connection to the clearing entity 137 periodically (e.g., every 60 seconds or so). If the connection is reestablished, the MQ manager 193 will transmit the tickets in the queue automatically. As shown in Figure 13e, the clearing entity 137 only messages back to the system 100 when tickets are received and accepted for processing. The MQ manager 193 relays the acknowledgment signal to the Back Office Adapter 112k. The Back Office Adapter 112k preforms a commit operation which updates the cache, database 111 and creates an AQ synchronization signal which is routed to the core RV ring, as shown in Figure 13e. As discussed above, the clearing entity 137 does not message back to the system if it receives a ticket but does not accept the ticket for processing. With reference to Figure 13a, the ticket monitoring tool 195 (e.g., a Broker Desk Tool) or the IBM 5250 emulation system 194 can monitor the status of tickets being transferred to the clearing entity 137.

The system 100 is equipped to handle problems associated with ticket transmission and acceptance between the system 100 and the clearing entity 137. For example, some ticket information may be problematic for the clearing entity to handle. This means that the clearing entity 137 received the trade but could not process the trade. In this case, an ICI Impact module residing at the clearing entity 137 location places the ticket in its repair queue. From the system's perspective no error was reported back from the MQ manager 193. In this case the Back Office Adapter 112k executes or changes the ticket status to exported. An exported status will be shown in the ticket monitor 195.

If the ticket was received by the clearing entity 137, but is problematic (e.g., needs intervention of some sort) it will be placed in the repair queue. For a broker monitoring the trade the fact that the status is not "accepted" can indicate that a problem exists. As discussed above, the ticket monitor can be used to determine whether the ticket has been accepted. Also the impact system can be accessed via terminal emulation to repair the tickets in the repair queue.

If the MQ manager's 193 link is inactive or otherwise unavailable, the MQ manager 193 will not signal an error and will continue to build its queue. In this situation, the connection between the MQ manager 193 and the clearing entity's 137 MQ manager is operational, but the process responsible for sending the tickets is not running correctly. Because an error is not reported back to the Back Office Adapter 112k, the ticket status is exported. The clearing entity system will not have any record that the ticket was received. From a monitoring perspective the fact that the status is not marked as accepted indicates that the above described problems exist. In this case the Impact module (or system) can be accessed via terminal emulation to see if the tickets are in the repair queue, and if not in the queue, it can be determine that a problem exists with the communications link.

If a ticket can not be transmitted, but there is nothing wrong with the clearing entity's 137 communication link, the MQ manager 193 issues an error back to the Back Office Adapter 112k. The Back Office Adapter 112k then changes the status of the ticket to broken. The ticket monitor as described above can be used to find "lost" tickets. If tickets cannot be transmitted because the connection between the MQ manager 193 and the clearing entity 137 is down, the MQ manager 193 will issue an error to the back office. In this case the Back Office Adapter 112k will mark the ticket as having a pending status. As in the above examples, the ticket manager 195 can be used to find tickets having a pending status.

Th MQ manager 193 and the Back Office Adapter 112k interact with other core components. For example, the MQ manager 193 adds messages to the queues from tickets passed on by the Back Office Adapter 112k. The ticket monitor tool 195 reads ticket status messages from the B-Desk application server. Also, a log file contains all trades for a given day (see Figure 13c). This file can be used to manually transfer trade information to the clearing entity 137 in the event that the transmission connection between the system and the clearing entity 137 fails. As discussed, the Back Office Adapter 112k accepts trade ticket sets and associated messages generated by the Trade Manager 112d and messages distributed via the core RV ring. The Back Office Adapter 112k is started and configured by the MCP 119 automatically when the system is started. Manual control of the Back Office Adapter 112k is accomplished using the MCP monitor. The Back Office Adapter 112k runs continuously unless it is stopped to facilitate system maintenance.

While the core components have been described with various individual functions, it will be appreciated by those skilled in the art that certain core components could be combined, or an individual core component could be configured to perform the function of another core component. Such modifications fall within the scope of the present invention.

Gateways

Gateway 113, shown in Figure 3, runs in its own domain on the server 110a located in the data center 110. As will be appreciated, the data center 120 has a gateway 123 operating on the server 120A. Optionally, as shown in Figure 1, additional gateways 171 and 181 can be located in remote areas. Gateways 171 and 181 are placed to limit the number of connections required for client connections to the data center 120. Without these remote gateways each client would need to connect directly to the data center 120. Numerous connections would be required without the gateways 171 and 181. However, gateways 171 and 181 provide service to multiple clients with less expensive local connections. In a preferred embodiment, the gateways 171 and 181 can be realized using a Sun Enterprise 6500/5500 server computer. The Sun Enterprise 6500/5500 can run the Solaris 2.6 operating system, from Sun Microsystems. The Sun Enterprise 6500/5500 server is manufactured by Sun Microsystems, Inc. Documentation for the Sun Enterprise 6500/5500 server can be obtained from Sun Microsystems, or through the above-mentioned web site.

One function of the gateways is to notify core components of order related transactions via a single commit operation, as discussed above, that simultaneously updates the gateway's 113's cache 118, database 111, and the synchronization stream. The gateways also interface with and forward system processing results to application servers, for example, servers 151, 152, 161, 162, 172, 173, 182, and 183. Gateways act a filters as they forward to a client only information that is applicable to that client or client site.

Gateway 113 interacts with other domains within the server 110a. For example, a commit operation performed by a gateway 113 after a service request by an application server simultaneously updates the system's database 111 and the synchronization stream. The data is then consumed by a particular core component that is interested in the data being transmitted. As inputs, the gateways receive information from application servers (e.g., 151, 151, 161, 162, etc.). In a preferred embodiment, the gateways 113 and 123 convert order request into objects representing the requests. The gateway 113 listens to the core RV ring to pick up RV messages resulting from core processings that need to be forwarded back to clients via the appropriate application server or connected gateway. Also, as shown in Figure 1, gateway 123 receives communication signals from both application servers 161, 162 and the remote gateways 171 and 181.

Gateways 113 and 123 are started and configured by the MCP 119 automatically when the systems are started. The gateways 171 and 181 in the remote locations 170 and 180, respectively, are configured in the same fashion. Manual control of the gateway 113 is facilitated with the MCP monitor. Under normal conditions the gateway operates continuously unless it is stopped to facilitate system maintenance.

Application Servers

The application servers (e.g., 151, 152, 161, and 162, etc.) runs at client sites 150 and 160. In a preferred embodiment, the application servers are Sun E450 servers. The Sun E450 server is a product of Sun Microsystems, Inc. Each of the application servers run the Solaris 2.6 operating system, or an equivalent format. Documentation for the Sun E450 server can be obtained from Sun Microsystems, or through its above-mentioned web site. As shown in Figure 1, the application server resides outside of the client's firewall 153 or 163, respectively. Each application server is connected to its respective data center via a external service network 114. Preferably application servers run in pairs. Using two servers at each client location facilitates load balancing and resiliency.

An application server accepts service requests from clients and updates its local cache. The application servers also forward service requests from clients to the gateway associated with the requests. The application server also receives transmissions from the gateway and updates to its power tier cache. As will be appreciated by those skilled in the art, the application servers can also be configured to perform information "pushes" to clients based on client interest. In this regard the application servers manages and tracks which clients are interested in certain information, and insures that the particular information is "pushed" to the appropriate client. As discussed, the application servers receive service requests from client/trader desktops and sends the service requests through the external network 114 to the gateway 113. Preferably, the application servers start automatically when the system is started or initialized. Application severs preferably run indefinitely, or until a scheduled shutdown or operator instruction to facilitate system maintenance, for example.

System Operation

The following discussion provides a general overview of an operational aspect of the Fixed Income Trading System 100. Although the following discussion necessarily implies a linear progression of events, it will be appreciated by those skilled in the art that the transactional nature of the system and the architecture make data updates and associated processing nearly instantaneously systemwide.

The following discussion will be given from the perspective of a trader associated with client site 150. As will be appreciated, however, the same or similar processing will occur from the client site 160 and data center 120. Furthermore, similar processing can occur from client sites 170 and 180, with additional handling through gateway 171 or 181.

The function and operation of the trader GUI software is described below with reference to Figures 23-40. The preparation of the trader GUI software would be readily apparent to a person of ordinary skill in the art, in view of this disclosure, preferably using Java Beans and the C++ programming language. As will be appreciated by those skilled in the art, Java Beans are software components that can be used in programing environments, like a visual programming environments. The trader GUI software runs directly on top of the workstation's operating system. Given that the trader GUI software is preferably Java-based, the trader GUI software is platform-independent and will run on operating systems such as Windows 98, other windows-based operating systems, UNIX, etc.

Traders can view "market activity" as well as executing transactions through the trader GUI. The GUI displays real-time market activity with respect to instruments posted on the system 100.

A log-in screen, as shown in Figure 23, is the first screen (e.g., dialog box) displayed to a trader upon launching the trader GUI software. Preferably, the trader sets up an account with the system offline (e.g. through a broker desk, or through an application process) and is issued a name and password at that time. Alternatively, an account set up process can be executed online to provide a user name and password for a trader. The system 100 (e.g., via the Session Manager 112h) validates a trader's user name and password by comparing the entered information against client information stored in the client database 135. Alternatively, user name and password information could be cached in the data center 110 or in the respective application servers. The Session Manager 112h, compares the user name and password information to see if they match. If the information matches, the Session Manager 112h relays an authorization message through gateway 113 to the GUI software on workstation 154. The message is transmitted from the gateway 113, through network 114 and application servers 151 and 152. If the authorization message is received, a trader gains access to the system 100. If the user name and password information do not match, the trader is denied access to the system 100. A password can be configured to be "trader site" specific. For example, the password and user name could only be used from a specific trade site. Preferably, only one instance of the GUI per user password can be operational at one time.

Upon entry to the system 100, the trader is presented with a main market screen 300, as shown in Figure 24. The main market screen includes a button bar 301, pull down menus 302, and three primary book views 303, 304, and 305. The main market screen 300 is fully configurable by the trader. For example, a trader can expand or compress the area of the displays or windows, and can determine which information is to be displayed through the views. In a preferred embodiment, the GUI uses a standardized color scheme to distinguish orders or requests on all display screens and dialog boxes. In one preferred embodiment, the following color convention is used: buys are represented by blue, sells are represented by pink, bids are represented by green, and offers are represented by yellow.

The market level (or "top of book") view 303 is a summary level view (both of bid and offer) of the instruments within a given market. For example, as shown in Figure 24, the trader has selected "U.S. Treasury" from the market level pull-down menu 306 to be displayed. An example of a market level selection screen is shown in Figure 27. Other market views could include European markets and U.S. Notes/Bonds, for example. The trader can also invoke other main market windows 300 to display types of instruments that the trader is particularly interested in. A trader can create a customized main market view by "clicking" on the market icon 312 in the button bar 301 in the recognized manner. The trader can select the market type (e.g., Euro-sovereign) or individual instruments to add to the new window 300. A tab is added to a tab section 307, which allows a trader to toggle between selected windows. "Clicking" on a tab (e.g., U.S. Treasuries, Short Books, ..., Euro-sovereign) on the customized views 307 displays the selected view. The GUI software accesses information associated with a particular view and displays the relevant information through the workstation 154. For example, with the selected market view shown in Figure 24, a market level view for all U.S. Treasury "books" (e.g., all U.S. Treasury instruments on the system 100 by settlement date) are displayed, along with a configurable window that permits the user to select the columns they want to display (e.g., bid, size, and offer).

One convention for U.S. Treasuries used by the system 100 and GUI software for "price" is dollars. Dollars are typically reflected in 1/8th of a 32nd. As will be appreciated by those familiar with trading fixed income instruments, there are also "symbols" which have specific "price" implications. For example, a US Treasury 10yr. note with a 98.28+ price, equates to 98 per 100 dollars, plus 28 32nd, plus ½ of a 32nd. Hence, the "+" equates to a ½ of a 32nd. Each security also has a market convention for amounts as reference is generally made to the total value of the security at maturity. In a preferred embodiment for U.S. Treasuries, an "amount" or "size" of 1 represents $1,000,000 par value at maturity. In the preferred embodiment with respect to U.S. Treasuries, a minimum amount increment is $1,000,000. As will be appreciated, the system can use other amount and price conventions depending on currency types, and can use different conventions for each instrument.

With reference to Figure 24, instrument 303a ("UST 6.000 15Aug09") has a "best" bid price of 95-18+, with an amount (e.g., size) of 352, and a "best" offer of 95-20, with an amount of 600. Multiple orders at the "Top of Book" price are aggregated. Meaning, that if there are three separate bid orders at a price of 99-04, each with an amount of 20, the top of book display would be a bid at 99-04 with an amount of 60.

The "Depth of Book" view 304 shows all orders (bids and offers, separately) that currently exist for each book within a market level. The depth of book 304 can be viewed by selecting a book from the top of the book or "key issues" views. For example, highlighting instrument 303a with a mouse or a keystroke in the standard manner, for example, would display all bids and offers associated with a book in the Depth of Book view 304.

A benchmark or "Key Issues" view 305 is a window displayed by the GUI software to show customized views of issues (e.g., benchmarks, sector, etc.). The Key Issues view 305 is linked to specific market levels such that instrument information is displayed each time the linked market level appears in the top of book view.

Order Submission

Order entry begins when a trader or user submits an order request from the trader desktop 154 to be posted in an instrument book, as shown in Figure 24. Traders populate instrument books by submitting orders, such as a bid request. To submit a bid, for example, a trader can "click" the bid button 310 from the main market view 300. Preferably, the trader is then presented with an "order entry" dialog box 320, as shown in Figure 25. Preferably, the information in the dialog box defaults to the instrument which is currently displayed in the Depth of Book view 304. For example, the information displayed in an instrument ID area 321 should correspond with the instrument displayed in the depth of book view 304. The bid price and amount can be changed via buttons 323 and 324, respectively. The order amount can be entered via the order amount area 325. The bid is submitted upon "clicking" the submit button 328.

Preferably, the trader has an option to set a system wide safety key on the GUI that activates a "confirm" dialog box for each order that the trader submits. The safety options are set in a "preference dialog box" 315, as shown in Figure 28. The preference box 315 can be activated from the Edit Tool, pull-down menu 302, for example. The preference box 315 also allows a trader to configure the Blotter 360, orders, market view, and to set default settings such as order size. The confirm box allows a trader to double check that the bid contains correct information prior to submission to the system.

An offer order is submitted by clicking the offer button 311, which activates an offer entry dialog box 350, as shown in Figure 29. The order entry box 320 can be completed in the same manner as was described above with respect to entering bids.

The order (bid/offer) is received by the application server 151 or 152 at the client site 150. The application server 151 or 152 updates its cache and delegates the request to the gateway 113 through the external network 114, as shown in Figures 14a and 14b. The gateway 113 creates the "object" necessary to process the order. For example, the object contains information regarding the order, routing information, and necessary processing information. Once the gateway 113 has created the order object, it performs a commit operation that simultaneously updates its cache 118, database 111, and the synchronization stream, as shown in Figure 14c.

A sequence number is assigned to each message in the synchronization stream as mentioned above. The synchronization stream is received by the AQP 112a component, which formats the messages into RV and broadcasts them into the core ring. With reference to Figure 14d, core components 112 consume messages from the core RV ring by subscribing to particular message subjects. In the present example, the Book Manager 112c recognizes the order request message in the synchronization stream. The Book Manager 112c consumes the message and updates its cache, but the commit operation does not write through to the database 111, since the data stream generated by the gateway 113 will also reach the database 111. The Book Manager 112c processes the order request. If the entered information is "valid," (e.g., the information conforms to the instrument's requirements or parameters) the order's object state changes from an "order request" to an "order." For example, the book manager 112c could check to see if a minimum amount and/or maximum order size are met for a given book. If the parameters are not met, the trader could be notified to enter the correct information. Size and price parameters can be established when the book is created, for example.

The Book Manager 112c also ensures that the new order is displayed and executed appropriately in a book. For example, for passive orders the sequence is price, time, then show. The best price order is always placed at the top of the book. At the same price, orders are sequenced on a first come, first served basis. In a preferred embodiment for the U.S. market, show amounts are executed before reserve amounts, as discussed below. In a preferred embodiment for the European market, orders with reserve amount have reserve amounts exhausted before shown amounts. Aggressive orders are processed on a first come, first serve basis. The Book Manager 112c could also be configured to execute orders in a market under a "price, time" rule, as is the preferred embodiment for a European market.

The Book Manager 112c then performs a commit operation that simultaneously updates the new information in the Book Manager's 112c cache, database 111, and synchronization stream, as shown in Figure 14e. With reference to Figure 14f, the Expiry Manager 112e and gateway 113 components receive the results of the Book Manager 112c processing via the synchronization stream on the core RV ring. The Expiry Manager 112e consumes the order message, updates its cache (again in a non-write through commit operation) and determines whether expiration criteria was submitted. This example assumes that the order should expire in five minutes. An order expire time can be set using the "order expire" dialog area 327, shown in Figure 25, for example. The gateway 113 receives the Book Manager 112c processing results and updates its own cache, as shown in Figure 14g. The gateway 113 then transmits the fresh book information to all applicable servers, as shown in Figure 14h. The gateway 113 also transmits client specific information about the order to the application server of the client that submitted the order. As will be appreciated by those skilled in the art, the application servers then push the information to the trader desktops (e.g., to the GUI software resident on the desktops) that have indicated interest in that particular book.

If the expiration elapses on the example order (e.g., the five minutes has elapsed), the Expiry Manager 112e sends an expiration request to the Book Manager 112c. The Book Manager 112c eliminates the order, updates the book, and the change is disseminated systemwide as described above with respect to a commit operation.

Order Acceptance

A trader can "hit" a bid (e.g., accept a bid to buy) by activating (e.g., "clicking" the HIT button 313, shown in Figure 24. Upon clicking the Hit button 313, the trader is presented with a Hit Bid screen 330, as shown in Figure 26. Here, the trader can enter an amount to sell, price and minimum amount to sell criteria for the instrument. Preferably, the Hit Bid screen 330 defaults to the instrument displayed in the Depth of Book view 304. A bid is submitted to the system when the trader clicks the sell button 333.

A trader can "take" or "lift" an offer (e.g., accept an offer to sell) by clicking or otherwise activating a "Buy" button 312. Upon clicking button 312, the trader is presented with a dialog box 340, shown in Figure 30. A trader can complete the take by filling out and submitting to the system relevant offer information, as discussed above with respect to "hitting a bid."

Hitting a bid (or "taking" an offer) is an aggressive order. In a preferred embodiment, an aggressor pays a system commission on an executed trade. The commission can be determined based on the size of the transaction, a percentage of transactions completed, or based on a flat rate, for example. From a system perspective, commission calculations preferably are handled by the Trade Manager 112d, for example, when processing tickets. Commission information can be displayed to a trader through a trader blotter 360 (e.g., Figures 36a-36c), discussed below.

From a system perspective, an object is created by the gateway 113 when the trader "aggressor" enters an order request. The object is forwarded to the Book Manager 112c, as discussed above. The Book Manager 112c validates the order request, creates an order, and checks the respective instrument book to see if there exists one or more orders having the same criteria. If the aggressor order has the same criteria as one or more passive orders, an execution is made. "Same criteria" means that a hit/lift has the same price, amount and/or instrument ID, as a bid/offer, for example. The processing results of the execution by the Book Manager 112c are picked up by the Trade Manager 112d and gateway 113. The gateway 113 transmits the effect of the execution to interested application servers systemwide. The Trade Manager 112d processes the execution data from the Book Manager 112c. The Trade Manager 112d performs post trade processing chores such as fee calculation, and creates tickets for settlement purposes. If this example occurs in the United States, each ticket has a $50 million dollar limit. In this case the Trade Manager 112d creates as many tickets as necessary to meet this requirement.

The system 100 produces tickets (e.g., via the Trade Manager 112d, as discussed above) so as to protect the client identity. A ticket reflects that the trade occurred with system 100 itself, that is to say, the system 100 is the contra side for each trade, as discussed below. Hence, a client remains anonymous. The Trade Manager 112d then commits the results to its cache, the database 111, and the synchronization stream. The synchronization traffic is recognized by the Back Office Adapter 112k. The Back Office Adapter 112k processes the ticket as discussed above with respect to the Back Office Adapter 112k.

From a system perspective, an aggressive order is processed according to established execution rules, as discussed above. The GUI software ensures that a trader has the order that they want to hit or lift visible for display (e.g., visible on the Depth of Book view) in order to process the order.

When aggressing an order (e.g., hitting a bid), a price option can be selected on the dialogue box to indicate either a "Limit or Better" 333a or "Average or Better" 333b, as shown in Figure 26. These options are used to determine which orders can be aggressed based on the price criteria set by the aggressor. A Limit or Better price option sets an absolute price limit (e.g., the system will not go beyond a specific price limit even if this means that the desired price and size cannot be achieved). An Average or Better price option is an average limit (e.g., the system will aggress orders below specified price limits in order to get closer to a desired amount, provided that the process of overall or average price remains better than the limit).

In a preferred embodiment, an individual's own orders are highlighted on the display. The GUI software prevents a traders from aggressing orders that were submitted by himself. Alternatively, the system 100 could be configured so that the Book Manager 112c rejects an aggressive order on a passive order that a trader has submitted himself. This feature prevents a trader from artificially moving a market, for example. Also, in a preferred embodiment, passive orders aggressed by a user within the same organization will generate a "ledger transfer." A ledger transfer is a notational record where no fees are exchanged, in lieu of tickets requiring settlement.

<u>Reserve Order Handling</u>

System 100 preferably permits a trader to hold a transaction amount in "reserve," and that reserve amount is not displayed system-wide to other traders. For example, as shown in Figure 25, a trader may reduce the "show" amount box 326 to show 10, which would automatically keep 10 in reserve. All reserve orders on the system 100 are "live"; meaning, that reserve orders will be used to fill aggressive orders. However, the extent to which reserve amounts are used to fill such orders can vary, as explained further below.

Preferably, shown orders are visible to all traders, while reserve orders are not shown to the market-at-large. A trader is able to view his own reserve amount through the GUI in a Depth of Book View 304. Holding a reserve amount enables traders to withhold from the market the full amount that the trader is willing to commit to the market. In one preferred embodiment, which for example can be used for European market trading, orders having a reserve amount are displayed systemwide with an "R" next to the shown amount. In a second preferred embodiment, which for example can be used for U.S. market trading, book views have no such indicator -- and thus both the existence of reserve, and its amount, are withheld.

As noted above, the extent to which the reserve are used to fill orders can vary. In one preferred embodiment for passive orders, the sequence is price, time, then show: shown amounts within a same price always execute before reserve amounts, and execution proceeds on a first come, first serve basis.

This methodology of satisfying an order is used in the preferred embodiment in U.S. markets. Of course, best price always predominates. Thus, if two orders exist at different prices, the order will be satisfied at the best price for the aggressor, utilizing both shown and reserve amounts, before using the next best order to satisfy the aggressor.

For example, as shown in Figure 31, four passive orders (Order Nos. 1-4) for an instrument (at the same price) are sequenced within a book as shown. The "show" column represents those orders that are shown system wide to all traders. The "reserve" column is an amount hidden, or held in reserve, from the market. The reserve amounts are preferably only displayed to the trader who submitted the order. If an aggressive order of 50 is received by the book manager 112c, Order No. 1's shown amount (i.e., 5) is executed first, and then Order No. 2's show amount (i.e., 25) is executed. Order Nos. 3 and 4's shown amounts are then executed to fill an amount of 45 for the aggressive order. An amount of 5 is then executed from Order No. 1 reserve amount to fill the complete order. In a preferred embodiment for U.S. markets, a trader's selected shown amount, once exhausted, is automatically replaced (or "filled") from the reserve amount, if any. "Filling" a shown amount will not result in a loss of any position, unless an order having a better price is received. Referred to as a "drip-rate," the reserve amounts automatically replace exhausted shown amounts, in one preferred embodiment.

In another preferred embodiment, the sequence for execution is price, then time: orders with reserves amounts have the reserve amounts exhausted before shown amounts. Both shown and reserve amount are executed moving on to the next order at a next price. For passive orders at a same price, orders are displayed on a first come first served basis. This methodology of satisfying an order is used in the preferred embodiment in European markets.

Modifying Orders

A trader can use system 100 to modify his or her order before it is hit or taken. Shown orders may be decreased without consequence to an order's ranking (e.g., listed sequence) in a book. Reserve orders may be increased or decreased without consequence, and this rule for modifying an order is preferably employed for U.S. markets. However, increasing a shown order amount in the preferred embodiment for U.S. markets will result in a loss of place in line if there are orders at the same price below the order. Preferably, the Book Manager 112c generates a message to be relayed to a trader that an order's ranking may be jeopardized.

Alternatively, the GUI software could evaluate an order and warn the trader, if needed, that his place in line for the respective book could be jeopardized. Of course, this is not an issue if the shown order is already last in line. An alternative option to increasing a shown amount is to enter a new order so that the original order amount retains its place in line.

In another preferred embodiment, increasing the overall size of an order, whether shown or reserve, results in loss of place in line. This rule for modifying an order is utilized by system 100 in European markets, where in satisfying an aggressor, reserve amounts are exhausted before moving on to the next order. Otherwise, a trader can potentially manipulate the show/reserve amounts.

Sweeping the Book

Sweeping the book is a function of system 100 that allows a trader to accept (e.g., hit/lift) multiple passive orders with a single operation. This function gives a trader the ability to hit bids or take offerings from the top of the order book down to a specific bid below the top bid price, or up to a specific offering price to accomplish a specific size trade. The sweeping function promotes quick risk management decisions. A trader can "sweep" the whole book, or part of the book, based on a maximum amount at a given price option. The orders are swept from the Depth of Book view 304 by placing the cursor at an appropriate starting point for the sweep. A "tool tip" pop-up dialogue box can inform the trader of the effective price and quantity prior to the sweep, based on shown amounts. For example, as shown in Figure 32, if a trader wants to sweep offers associated with a given instrument, he activates a "sweep offers" 350 dialogue screen. Using dialog screen 350, the trader can select the amount of the instrument, offer price, and pricing scheme. With respect to the pricing scheme, a trader can select either a "Limit or Better" 353a or "Average or Better" 353b price option, as discussed about with respect to hitting a bid. Sweeping bids is achieved in the same manner. The sweeping-the-book function increases transaction speed and efficiency.

From a system perspective, the Book Manager 112c receives and executes the "sweep" order based on the price and amount criteria in the order request. The executed orders are received by the Trade Manager 112d which creates tickets. A sweeping transaction is viewed as a single execution from the perspective of the aggressor, since the system 100 is the aggressor's counterpart. Hence, if n orders are swept, n+1 tickets are created by the Ticket Manager 112d.

In a preferred embodiment, the Book Manager 112c follows the same execution rules discussed above. Furthermore, in the preferred embodiment, the Book Manager 112c executes reserve amounts at a given price, after executing shown amounts, and before executing at a next price level. Also, in a preferred embodiment, when a trader sweeps the book, and as a result aggresses orders that are lower in the book than one or more of his own passive orders, the system will expire the trader's passive orders.

<u>Residual Posting</u>

A residual amount is defined as an unfilled balance of an aggressive order. Through the GUI, a trader has an option to authorize the system 100 to post a residual amount as a passive order in the subject instrument book. A trader can post a residual by "checking" a box in each of the hit/bid 330, take offer 340 and "sweep" order (e.g., 350) dialogue boxes, as shown in Figures 26, 30 and 32, respectively. If a residual box is checked by the trader when entering the order, the system will post the residual at the aggressed-on price; meaning, that residual orders will be posted at the highest bid/lowest offer aggressed. In a preferred embodiment related to the U.S. market, residual amounts are posted entirely as shown amounts, with no reserve listed. In a preferred embodiment related to the European market, residual amounts are divided between a minimum size for a book show amount, and the remainder is stored as a reserve.

Once a "Post Residual" box is checked, it preferably remains in effect until the trader removes the check from the box. Residual amounts less than a book's trade size minimum will be displayed if they are "Top of Book". Preferably, to aggress on these "below minium amount" orders, traders combine at least part of another passive order to meet trade size minimums. In a preferred embodiment, any residual amount less than 5 will be canceled automatically once it is no longer a top of book order.

From a system perspective, the Book Manager 112c executes orders having a post residual authorization as discussed above. If an order contains such authorization, and a residual amount is realized from executing the order, the Book Manager 112c converts any residual amount into a passive order to be listed at the aggressed-on price in the respective book.

Allowing a trader to post a residual provides efficiency to transactions, since a trader will not have to reenter the unexecuted or unfilled amount.

Quick Hit/Quick Take

Through the workstation GUI, a trader is provided with an option to execute a Quick Hit or Quick Take order, as shown in Figures 33 and 34, respectively. A quick hit or quick take box defaults to the settings that are selected in the preferences box 315, shown in Figure 28. As a result, a trader can increase transaction efficiency, by reducing order entry time. In essence, the Quick Hit/Quick Take function is a hit or take "shortcut."

In a preferred embodiment, the Quick Hit or Quick Take dialog box can be activated, for example, by right clicking on an instrument listed in the Market Level view 303, Depth of Book view 304 or Floating Book view 370. Like a regular order, an object is created to represent a Quick Hit or Quick Take order. The Quick Hit or Quick Take request order is created and processed by the system (e.g., initially by the gateway 113 and Book Manager 112c), as described above.

Good-Until-Expiration

Through the GUI, a trader can select expiration criteria for each passive order. As shown in Figures 25 and 29, a "Good Until" drop down menu 327 or 357, respectively, allows a trader to set an order's expiration time or date. For example, expiration criteria includes "Good Until Topped," "Good Until Time," "Good Until End of Day," and "Good Until Canceled." The default expiration setting for all orders is Good Until End of Day. The Good Until End of Day option automatically expires orders which have not executed by trading day's end. An expired order is deleted from its respective book.

When the Good-Until-Topped function is set, an order remains in a book as long as the order is the best priced order in its respective book. A Good-Until-Topped order will remain active until another order is received that "tops" it. That is, the Good-Until-Topped order expires, and is deleted from the respective book, as soon as an order having a better price is entered into the instrument book. In a preferred embodiment, the Good-Until-Topped order also expires at the end of a trading day. At that time, the order is treated as a Good-Until-End of Day, as discussed above.

The Good-Until Time function allows a trader to enter a specific expiration time or duration (e.g., 15 minutes) for an order. For example, Figure 25 illustrates that the expiration time is set to be 3:41 p.m. If the bid is not hit by 3:41 p.m., it will expire. A Good-Until-Canceled expiration function keeps the order in the system until it is canceled by a trader. This type of order will not expire at the end of a trading day, but will remain active until it is executed, or until it is canceled by the trader.

From a system perspective, expiration settings are processed by the Expiring Manager 112e, in connection with the Book Manager 112c, as discussed above.

<u>Minimum Order Execution</u>

For aggressive orders (e.g., hits or takes), system 100 preferably permits a trader to set a minimum transaction size required for an order execution. For example, a trader can "hit" passive bids (at a selected price option) for an amount of 275, but will accept a minimum transaction of 25 (see Figure 26, reference numbers 331 and 336, for example). Thus, if the order book contains one or more orders at the requested price totaling 25 or more (including reserves), the aggressive order will be satisfied. However, if the order book does not contain orders ta the requested price totaling at least 25, then the aggressive order will not be satisfied.

Pinging

Using this minimum order execution feature, a trader can obtain information about the reserve amount for a given price option. A failed aggressive order of this type does not alter the other side (or sides). The other side will only be notified if an execution occurs. At the same time, the Book Manager 112c preferably generates a failed notification, to be relayed to the trader, in the event that the minimum order amount cannot be filled. Thus, if an order is not filled, the aggressor is able to determine that the existing orders at the requested price do not have reserves at least equal to the minimum order amount less the shown order amounts (at the requested price).

Auto-Aggress

An inverted market occurs when a bid is submitted with a higher price than the best offer, or an offer is submitted with a price that is less than the best bid price. To address an inverted market, the system 100 (e.g., the Book Manager 112c in one preferred embodiment) preferably has an Auto-aggress feature that automatically converts the inverting passive order into an aggressive order. The newly converted aggressive order executes at the best bid/best offer on the system. The system 100 automatically posts as a passive order in the respective book any residual amount at the specified price.

In a preferred embodiment, a trader that inverted the market will pay a commission fee, since that trader's order is treated like an aggressive order. Also, in a preferred embodiment, a passive order that inverts another passive order from the same trader is rejected by the system 100.

Figure 35 is a flow diagram illustrating one preferred operation of the auto-aggress feature, as discussed above. In step S1, the book manager 112c receives passive orders requests (e.g., objects representing passive order requests), as discussed above. In step S2, the book manager 112c compares the passive order requests against current requests (if any) in the book. The book manager 112c determines whether the passive order request inverts another passive order from the same trader (step S3). If the order request does invert another passive order from the same trader, the order request is rejected by the book manager 112c (step S4). The trader is notified that the order request has been rejected in step S5. If the passive order request does not invert another passive order from the same trader, the Book Manager 112c determines in step S6 whether the order request inverts the market. If the order does invert the market, the book manager 112c converts the order request into an aggressive order in step S7. The Book Manager 112c then executes the order in step S8 as discussed above. The trader is notified of the executed order in step S9. If the order request does not invert the market, the book manager 112c processes the order and lists the order in the book (S10). Preferably, the trader is then notified that the order has been entered into the book in step S11. As an optional step, the system could notify (e.g., via a pop-up dialog box) the trader that he is about to invert the market, and verify whether he wants his inverting passive order to be converted into an aggressive order.

Contingent Trading/Pegged Trading

Contingent trading is the process by which one can link a simultaneous execution on a buy and a sell transaction between two securities that are specified by the trader at a specific price or yield spread basis. The system 100 facilitates contingent trading through the GUI. This type of execution, which populates two order books, is contingent upon both price specifications being realized.

Pegged trading is the ability of a trader to specify the price behavior of one security on the system 100 to the execution of the other security. For example, a trader may want to sell a 30-year bond when the 10-year bond achieves a specific price bid. Again, the GUI notifies and the system 100 automatically makes these types of transactions when the specified criteria is reached.

Pegged Bids and Offers

A "pegged" market can be defined as a bid or offer on a specific security that uses another security or index as a benchmark. In a preferred embodiment, pegged trades are executed automatically by the system 100 when the current market on the system 100 reaches a "correct" price. The correct price can be determined by a new price in the pegged instrument, which generates a new bid or offer and/or determined by a "spread" between the security to be executed and the pegged instrument.

Pegged bids or offers are dependent upon the price of a bid or offer for an independent security. The price can be expressed as a "spread" to an independent instrument. In effect, every time the Top-of-book bid or offer price on the independent instrument changes, so does the dependent bid or offer.

The independent instrument for pegged bids and offers is preferably an instrument for which a book exists on the system 100. However, it is possible that no bids or offers may be listed (or one active) in that book. In that case, the pegged order will not be active until an order exists to peg the price. If previously active, a pegged order will be automatically suspended if the independent top-of-book is not available.

A trader specifies whether they are pegging to a Top of Book bid or a Top of Book offer. In comparison, contingent orders are preferably bid to bid(s) or offer to offer(s).

An "inverted market" could occur as the result of a price revaluation on a pegged bid or offer. In these cases, the system 100 preferably does not show a bid price higher than the existing offer (conversely, a bid yield lower than the offer yield). Preferably, the system 100 automatically execute bids or offers under the following circumstances, particularly when sufficient liquidity exists at the Top-of-Book. First, a bid revalues such that its price is greater than the Top-of-book offer price plus commission. In other words, lifting the offer to buy and paying commission is more advantageous than bidding. In this case, price-improvement goes to a trader who originally entered the pegged order. Second, an offer revalues such that its price is less than the Top-of-book bid price minus commission. In other words, hitting the bid to sell at a price plus commission is more advantageous than offering. In this case, price-improvement goes to the person who entered the pegged order.

The following example is provided to further illustrate pegged trading.

Example 1 – Pegged Trading

Client XYZ enters a pegged bid for Security B of 10 million with the bid price pegged to Security A plus 1-027. 1-027 could also be expressed as 1 plus 2/32 plus 7/8 of 32 or 1 plus 2 7/8 32nds.

For this example, Client XYZ is assumed to be the only bidder for Security B. As discussed above, the "aggressor" preferably pays a commission for trades on the system 100. Furthermore, for this example, Client XYZ does not "go free" on commissions (i.e., XYZ always pays if it is the aggressor). For illustrative purposes only, XYZ pays less than 1/8 of a 1/32nd commission.

Event 1: Client XYZ enters the following bid:

Security A			Security B		
Bid	Offer	SIZE	Bid	Offer	SIZE
98-10	98-124	10X20	99-127	99-134	10X50

The Bid on Security B is calculated as 98-10 plus 1-027 = 99.127. At this price, the bid on Security B would not be executed, because it is still lower than the best offer (99.134). In this example, SIZE = 10x50 means the bidder will take up to 10 Million Par at that price, the offerer will sell up to 20 Million Par at that price.

Event 2: The Top-of-Book Bid Price on Security A changes to 98-101 (e.g., the price of A goes up by 1/8 of a 1/32nd)

Security A			Security B		
Bid	Offer	SIZE	Bid	Offer	SIZE
98-101	98-123	10x20	99-13	99-134	10x50

The result of the bid price on Security A changing is that the bid for Security B also changes (e.g., 99-127 plus 1/8 of 1/32 = 99-13). However, there is still no trade executed because 99-13 is less than 99-134 by 1/2 of a 1/32nd.

Event 3: The Top-of-Book Bid Price on Security A changes to 98-105 (e.g., the Bid Price on A goes up by 1/2 of a 1/32nd).

Security A			Security B		
Bid	Offer	SIZE	Bid	Offer	SIZE
98-105	98-124	10x20	99-134		10x50

The result of the bid price on Security A changing, is that the bid for Security B also changes (e.g., 99-130 plus 1/2 of 1/32 = 99-134). However, this change still would not result in an automatic execution, since XYZ would pay more than the "spread" once commission is calculated. XYZ would typically not consider itself an aggressor in this case. Hence, the trade will execute when the bid price is greater than or equal to the offer price, plus commission. In this case, the bid of 99-134 is not greater than 99-134, plus the commission. A situation where the bid price is equal to the offer price, and a trade can not execute, is referred to as a "locked market."

Event 4: The Top-of-Book bid Price on Security A changes to 98-106 (i.e. the Bid Price on A goes up by 1/8 of a 1/32nd)

Security A			Security B		
Bid	Offer	SIZE	Bid	Offer	SIZE
98-106	98-124	10x20	99-135	99-134	10x50
98-106	98-124	10x20	99-134	99-134	10x50
98-106	98-124	10x20		99-134	x40

The first line with numbers under Event 4 represents a calculated Bid/Offer, the second line represents prices used for execution, and the third line represents how the market will appear after the trade executes.

As a result of the market bid price of A going to 98-106, XYZ's Bid Price for Security B goes to 99-135. The bid price (99-135) in this example is greater than the Offer price (99-134) plus commission (less than 1/8 of a 1/32nd). In this case, XYZ would be willing to pay the offer price plus commission (since it is assumed that the commission will be less than 1/8 of a 1/32nd). Accordingly, a system-initiated trade for a pegged buy order of 10 will execute at 99-134. As a result, XYZ will pay 99-134 plus a commission. The trader on the Offer side will be paid 99-134 and pay no commission. After the trade executes, the amount offered on B is reduced from 50 to 40. After the trade executes, there are not other bids for Security B (according to the assumptions). Preferably, only the final line under Event 4 would be displayed through system 100 as a result of the trade.

As will be appreciated, in Event 3 the result was a locked market. If XYZ was a trader did not pay a commission per execution (i.e., they are allowed to "go free" or he pays a flat fee of some sort to the system), then the trade would have executed. In other words, if the commission charged to XYZ were zero then the matched bid and offer would generate a transaction. In this case, the bid of 99-134 would be greater than or equal to the Offer of 99-134 plus the Commission of 0 (99-134 = 99-134).

Also, if XYZ's commission rate is greater than 1/8 of 1/32, then the trade would still not execute in Event 4. In this case, the bid of 99-135 would be less than the offer of 99-134 plus the commission. This would appear to create an "inverted" market in which the bid price (99-135) was higher than the offer price (99-134). If this occurred, the system 100 could either not display the 99-134 bid vs. 99-134 offer, or could deliver a message to the trader regarding the inverted market scenario which would explains why his bid is not being executed.

As an alternative example, another commission paying trader, ABC, enters a bid of 99-134 for Security B. This creates a situation where Security B is "locked" for the same reasons it was locked in event 3. When Event 3 occurs, XYZ will just be one more bidder at 99-134, which can not execute. However, when event 4 occurs, an execution will take place and XYZ will appear to obtain Security B at 99-134 despite the fact that ABC was there first with the same bid. XYZ's bid will be filled because in reality they are paying a higher price than the posted price due to the inclusion of the commission. ABC will go without an execution in this case.

As will be readily apparent from the foregoing discussion, the Book Manager 112c preferably manages the pegged orders and performs trade executions, notifications and book management, as has been discussed in view of a pegged trade.

Contingent Bids and Offers

Contingent orders are bids and offers that require two opposite trades to be executed simultaneously when the price differential between the two instruments reaches a specific level.

A "single aggressor" scenario will be described. A single aggressor scenario involves an order being entered with a "spread" between a dependent and an independent security. Similar to Pegged Trades, an order is placed in the book of the dependent security and is revalued based on bids or Offers in the independent security's book. In addition to the spread, the trader specifies the amount of the independent security that should be accepted. If and when the order in the dependent book is accepted, the system automatically accepts the order in the independent book. It is in the independent book where the user, due to being an aggressor, is obligated to pay a commission.

The single aggressor model preferably requires that orders be placed such that bids are dependent upon other bids, and offers are dependent upon other offers.

It is possible that no bids or offers may be active in the independent book or that the total of the orders in the book are not of sufficient size to fill the contingent order. In that case, the contingent order will not be active until an order exists to value the price of the order in the dependent book.

An "inverted market" could occur as the result of a price revaluation on a contingent bid or offer. In this case, the system preferably does not show a bid price higher than the existing offer (conversely, a bid yield lower than the offer yield). Like a case of pegged trades, it may be possible for the system to automatically execute bids or offers when this occurs.

Contingent orders differ from pegged orders in that they are preferably bid to bid(s) or offer to offer(s).

The following examples are provided to further illustrate contingent trading. In examples 1-3, a quote for a particular instrument *x* is referred to as Q*x*, where *Qx* is a body of information or function of three variables {Y*x*, V*x*, P*x*}. In examples 1-3, V*x* is a volume, par amount, or size of the bid or offer, Px is a dollar price, Yx is a yield

<u>Example 1 - (SINGLE AGGRESSOR)</u>

In this example a pair of instruments "x" and "*y*" are at issue. A trader A acts as an aggressor for instrument y, after A's order on *x* has been acted upon aggressively by another trader B. For this example, *y* is the more liquid of the two instruments and will be the independent instrument within the pair. Instrument x is the instrument dependent upon Y*y* or P*y* of *y*, and Trader A will not be the aggressor in this security, but enters an order on *x* to buy or sell at a level dependant upon the Y*y* or P*y* of *y*. This order on P*x* and P*y* will be linked by some relationship determined by Trader A and this relationship can be expressed, for example, as a dollar price difference, price ratio or yield spread between B*x* and B*y*.

Vx and Vy will also be determined by Trader A and may or may not be equal (face value on both sides), or could be determined by a duration or risk equivalent calculation. This type of transaction can occur in the normal continuous trading environment. As an example, Trader A wants to sell x and buy y according to a specific relationship.

Trader A would enter an order into the system 100 (e.g., through the trader GUI) to sell x with the requirement that the sell is "contingent" upon buying y. If the system 100 detects an offer for y expressed as Qy, an offer for Qx is preferably automatically posted by the system for Trader A according to the relationship designated by Px and Py. If this offer, Qx, is lifted by a third party, the system will automatically and simultaneously lift the offer Qy on behalf of Trader A, before allowing the transaction for the selling, by Trader A, of *x*. If *y* fails to transact, then *x* will fail to transact. The contingent trade will thus be transacted with Trader A acting as aggressor in B*y* where he will be paying a commission.

In this transaction, Trader A has sold *x* and bought *y on a* price contingent basis. If the offering on P*y* moves, the system 100 automatically adjusts Q*x* accordingly. If the offering on P*y* disappears or is withdrawn, the offering on Q*x* is automatically withdrawn. If the offering on P*y* is re-instated, the offering on Q*x* is automatically re-instated. If V*y* is does not permit partial fills, Q*x* should have notice to this condition and can only be displayed within the restrictions linking V*x* and V*y*. In a preferred embodiment, contingent trades are reflected as such through the trader Blotter 360. An accumulation of separate trades as part of an overall contingent order may be treated as one order for commission purposes.

Example 2 (NO AGGRESSOR)

In this contingent trade example, a trader does not act as an aggressor on either side of the transaction, but merely enters his requirements as a condition linking Qx and Q*y*. This type of trade could be referred to as a 'package' or 'switch' trade. Typically, this type of trade would only be transacted if an aggressor had the exact opposite transaction to perform. When the aggressor enters the request to accept the Q*x* he would be given an indication that a simultaneous transaction at Q*y* must occur in order for his request to be processed. Preferably, the trader is prompted through the trader GUI to provide Q*y* to become a contra party for that transaction, or to decline to provide Q*y*. If the trader declines, the system 100 could search for Q*y* and only process Q*x* if it was found. This type of trade would be entered in the Book Manager 112c as a function of Q*x* and Q*y* (e.g., "f{Q*x*,Q*y*}"). In one preferred embodiment, this type of contingent trade could be separately displayed as a non-aggressor quote driven contingent trade.

In a typical book view, as shown in Figure 24, instruments *x* and *y* could be "flagged" as also appearing in a "contingent trading page" (e.g., a separate view or screen). Traders with an interest in either x or y could gather more information from viewing the contingent page. In this example, contingency may refer not only to price or yield conditions, but also to settlement and delivery conditions. For example, if one party does not deliver B*x* then he does not have to accept delivery of B*y*.

Example 3 (DOUBLE AGGRESSOR)

In this contingent trade example, a trader acts as an aggressor on both sides of the trade, and instructs the system 100 to "capture" posted bids and offers if they are available. For example, an initiating trader may want to sell *x* and buy *y* on a fixed yield spread which is a calculated function of the prices for these bonds, call it f{P*x*,P*y*}. In this example, the trader can enter his requirements into an order entry dialog box, or through the trade blotter 360, and the system automatically and continuously scans available quotes in the two relevant securities. If a bid (P*x*) and an offer (P*y*) which satisfy the conditions f{P*x*,P*y*} are found, the system automatically executes the transaction on both sides simultaneously and perform the contingent trade.

For example, if Trader A is willing sell x at a 5 point spread to y, and y is offered at 4.95%, the system will float the order. The system will show an offering on x at 4.90%. If a third party enters an order bid at a 4.90% on x, and y is still offered at 4.95%, the system 100 would buy y and hit the bid at 4.90%. The system 100 would then sell x for Trader A. The bidder receives their execution on x at their level of 4.90%, and is preferably unaware (e.g., not notified) of the simultaneous transaction on y that has been system 100 generated.

As will be appreciated, the system 100 would only execute automatically if V*x* and V*y* were available in the relevant (duration weighted) amounts. This will usually mean that partial fills will need to be permitted on one side of the transaction or the other. This type of contingent trading is particularly useful when dealing with large numbers of securities over several screens in the system 100. On occasions a trader may only wish to be alerted when quotes are available which would enable the contingent trades on the desired terms. In this case, the system 100 could notify a trader through the trader GUI of the availability of such a contingent trade.

Example 4 – Linked Trading

A linked transaction is a system 100 ability to simultaneously trade two or more securities linked to a spread, allowing for the bid/offer spreads between the issues. In this example, the following information is available regarding Securities A and B.

Security A	Security B
Market on System 100	Market on System 100
98-09 x	-10 99-11 x -12
5.959 x	5.948 5.941 x = 5.929

Trader 1 wants to enter an order to sell security A, but linked to a transaction to buy security B. Trader 1 would like to execute this order regardless of the dollar price at a spread of .2 basis points. In this example, 1 basis point is equal to ten one hundredths of a percent(%). At the current market on the system 100, a sell on security A would execute at a yield price of 5.959, and a buy of security B would execute at a yield price of 5.929 which is a spread of (5.959-5.929) = .03. As will be appreciated, since a basis point is reflected in ten one hundredths, .03 is .3 basis points. Given the current market on the system 100 as shown, Trader 1 cannot execute his trade of a 0.2 basis points spread.

In another example, the system 100, after the Trader 1 has entered his .2 basis point order to sell A and buy B, would generate a bid on Security B at 99-115 to yield 5.939 for as long as the bid on security A is 98-09 to yield 5.959. The system 100 would also generate an offer on Security A of 98-096 to yield 5.949 for as long as the offer on security B is 99-12 to yield 5.929.

In this example, three events are discussed. First, the trader buys security B at his system generated bid. If he buys security B at 99-115 the system sells security A at 98-09 and pulls the offer on security A at 98-096, which results in the execution of his .2 basis points linked transaction. Second, the trader could sell Security A at his system generated offer. In this event, if he sells security A at 98-096 the system buys security B at 99-12 and pulls the bid on security B at 99-115, which results in the execution of his .2 basis point linked transaction. Third, nothing happens and the trader's order floats as the market price changes on security A and security B.

Negotiation

The GUI also allows a trader to indicate whether or not they are interested in negotiating the terms of a particular instrument. For example, as shown in Figure 25, a dialog box 329 is provided to indicate that a trader is interested in negotiating the terms of the particular security. If an aggressing party is interested in a particular passive order, they can notify, through the system 100, the trader who entered the passive order, and conduct on-line or off-line negotiation. As will be appreciated by those skilled in the art, if the negotiation occurs on line, both parties can access a chat room or an instant message negotiation, for example.

Another aspect of negotiation is to allow a trader to privately and anonymously negotiate the terms of an execution. In a preferred embodiment, a trader activates a "negotiated order" dialog box through the trader GUI, to sends an order to any active bid or offer on the system. The negotiated order can include desired transaction criteria, such as order expiration time, piece specifications (minimum, maximum, specific size only), settlement date, etc.

The target trader preferably can view and filter such private negotiated orders, through, for example, the blotter 360. To respond to a negotiated order, the target trader can Hit the "bid" of the negotiated order. In this case the target client would be considered the aggressor and would be responsible for any commission. Such a transaction could also result in a reduction in the size of the order placed by the target client by the amount traded. The target client preferably has the option of keeping the "public" order at the same amount or modifying the size of the order, for example.

The target trader could also respond by sending the initiating trader a new "negotiated order" with a lower price, for example. The initiator trader could then respond as suggested above. The target trader could also respond by pulling their order, modifying the size of their order, or simply letting the order expire.

As will be appreciated, it is also possible that in the middle of the negotiation session that the target client's bid or offer could be executed. At that point the target trader is preferably notified by the system 100 that the subject order has been executed.

ADDITIONAL SYSTEM FEATURES

Blotter

The GUI provides a "blotter" 360, as shown in Figures 36a-36c, of all trades that the trader has executed. The blotter 360 is a personalized work space that displays only the trader's own market activity. The blotter 360 filters information related to the trader, including, bids and offers, time (all trades within a specific time), price (all trades within a specified price), and amount (all trades with a specified amount), for example. From the button bar 301 (Figure 24), a trader can activate the blotter 360, as shown in Figures 36a through 36c. The blotter 360 can display top of book information, for example. From the blotter 360, a trader can view order information by activating tabs 361. Such information may include the trader's orders, including, executed, suspended, expired, canceled, and rejected orders. For example, as shown in Figure 36b, activating a "Rejected Request" tab indicates whether a trader has had any requests rejected. Also, as shown in Figure 36c, a trader can activate the "Suspended" tab to see whether any orders are currently suspended. Like the main screen 300, a trader can access a particular market or book through a button bar 362. A trader can also suspend or cancel orders from the blotter 360 screen.

The blotter 360 serves as a customized filter in that it displays information pertaining to the trader's orders and transactions. As will be appreciated by those skilled in the art, the GUI gathers information (e.g., in one embodiment, from objects related to books, orders, requests, tickets, etc.) that is cached in the application servers, or that is pushed from a data center 110 or 120. The gathered information pertains to the trader's own market activity within the system 100.

Floating Book

To free up desktop space, a trader can choose an instrument of interest (e.g., instrument 303a in Figure 24), and then activate a "floating book" 370 for that instrument by clicking on icon 311 on Figure 24. Figure 37 depicts a floating book for a particular instrument. A floating book 370 can be displayed independently of the main screen 300; meaning, the main screen 300 can be opened or minimized (or closed), while a floating book 370 remains open. Preferably, a trader can concurrently open many floating books, each representing separate instruments.

A trader can submit hit, bid, offer, or take orders from the floating book 370 screen. A trader can also activate the Quick/Hit or Quick/take feature from the floating book 370 screen by right "clicking" on a highlighted order and selecting the Quick/Hit or Quick/take from an activated drop down menu.

Indicating Last Order

The GUI software displays the last trade that was executed for each book. As discussed, last trade information can be included in the Depth of Book view. For example, as shown in Figure 24, the last executed trade for the instrument shown in the Depth of Book view 304 is displayed in area 309. The last trade feature provides a trader with real-time information regarding a particular instrument.

From a system perspective, the book manager 112c can update a book when a trade is executed, as discussed above, by including last trade information. The last trade information can be distributed throughout the system (e.g., through a normal "commit operation" and gateway message forwarding, as discussed above). As will be appreciated by those skilled in the art, the GUI could then display the last trade information. Preferably, the last trade information can also be viewed from a floating book, as shown in Figure 37.

Suspend/Cancel/Modify Orders

Through the GUI, a trader can cancel, suspend, or modify orders that he has submitted. For example, a trader can modify orders that he submitted from the Depth of Book view 304 shown in Figure 24. A trader can modify an order's price, quantity, and reserve amount terms for a particular order. In a preferred embodiment, a trader's own orders are highlighted (e.g., are displayed in white or are bolded) on the various book views. A trader can modify his order by placing the cursor on the target order, right "clicking", and selecting a "modify" option from the drop down menu. A dialog box (now shown) is activated to facilitate any changes to the order. Orders can be modified from the blotter, in a similar manner. Preferably, a warning is issued to the trader if a size modification could potentially change the priority of the order.

Orders can also be suspended through the GUI. For example, as shown in Figure 24, the button bar 301 provides a "Suspend Orders" button. The Suspend button allows a trader to suspend a single order or suspend all orders that have been submitted to the system by that trader. Preferably, a "Suspend All" feature is invoked each time a client logs off the system or is inadvertently disconnected from the system. Orders that are suspended are preferably removed from a book. From a system perspective, the Book Manager 112c deletes or otherwise drops the order from a book and then updates the system through a commit operation as discussed above. Preferably, however, the suspended order is moved to the "suspended" section of the trader's blotter 360 as shown in Figure 36c, and is stored locally on the trader's workstation or in the application servers. Preferably, each suspended order is individually reinstated by clicking, for example, on the order from the suspended blotter as shown in Figure 36c. Alternatively, groups of suspended or canceled orders may be reinstated in bulk.

The system treats a reinstatement as a new order request. Consequently, the former suspended order will appear in the book as a new order. From a system perspective, reinstating a suspended order is handled as a new order request, as discussed above. A trader can also suspend all live orders from the blotter view 360. As shown in Figures 36a through 36c, a suspend-live-orders dialog box 363 is provided for the trader's convenience. Clicking or otherwise activating the "O.K." button 364 will suspend all live orders.

A trader can also cancel an order or all active orders by clicking on the cancel live orders button as shown in Figures 36a through 36c. A trader can also cancel orders from the main view 300 as shown in Figure 24 by activating the "Cancel Orders" button on the button bar 301. Preferably, a trader is then presented with a Cancel Orders dialog box 380, as shown in Figure 38. The Cancel Orders box 380 permits a trader to cancel a particular instrument, or cancel all orders, bids or offers, for example. The trader can also select which market level to choose from. Like the suspend feature, order cancellations can be invoked on an individual order basis or in a bulk basis. Preferably, a bulk cancel feature is invoked for a particular book, whenever that book is "closed" (e.g., typically a book is closed at the end of a trading day). From a system perspective, the Bulk Request Manager 112g receives and processes "suspend all" or "cancel all" requests. The Bulk Request Manager 112g forwards an appropriate message to the Book Manager 112c, which in turn deletes or otherwise drops the orders from their respective books, as discussed above.

A trader may only modify, cancel or suspend orders that originally were entered by that trader. Preferably, an exception to this rule is that if a broker is acting on behalf of a trader, then the broker will be able to perform any updates that the trader would have been able to perform had they logged off from the system as discussed below. The features of system 100 that permit a broker to act on behalf of a trader are described in the "broker tools" section below.

<u>Notification</u>

Figure 39 illustrates an Execution Notification window. Through the Execution Notification window, the system 100 notifies a trader when a trade has executed. In this way, a trader can be confident that his trades have executed. From a system perspective, the Book Manager 112c issues a trade execution to the Trade Manager 112d, and at the same time updates its cache and issue an AQ message to the system. This AQ message is relayed by a gateway to the trader's client site. As will be appreciated by those skilled in the art, the GUI software displays this information to the trader.

<u>Multiple Depth of Book Views</u>

Through the trader GUI, a trader can configure a window display to show multiple Depth of Book views, each for separate instruments. Preferably, a trader activates the multiple view screen through an option in the market 302 pull down menu shown in Figure 24. Figure 40 illustrates a window screen 390 having four depth of book views concurrently displayed on one screen. Preferably, a trader can configure several multiple view displays, which can be accessed through tabs 391. A multiple view tab is also added (preferably automatically) to the tab section 307 shown in Figure 24. A trader can access his blotter 360, market levels, and suspend or cancel orders through the multiple depth display 390, as shown in Figure 40.

Fixed Income Ticker

Through the GUI, a trader can activate a moving "ticker" 308 that presents real-time information regarding instruments trading on the system 100. Preferably, a trader can customize the ticker to include those instrument information that the trader is interested in. In a preferred embodiment, a selected instrument (and related trading information) moves across the ticker area 308a. As will be appreciated by those skilled in the art, the GUI displays relevant information from an instrument's book for the moving ticker display, which object processing is preferably handled by the associated application server.

Process Bar

The GUI also includes a process bar 310, as shown in Figure 24, that is accessible from the main screen 300. The process bar 310 provides a running list of all orders or requests submitted by the trader. The process bar 310 also provides a status indicator. The status indicator displays to the trader the status of each order. For example, when the Book Manager 112c has received and processed an order (e.g., updated the relevant instrument book to include the order, or executed a trade), the "processed" status is display, as shown in Figure 26. Other status indicators may include "accepted," "canceled," "suspended," "executed," and "rejected," for example. With these features, the process bar 310 provide yet another efficient means for a trader to monitor his trading activity.

As will be appreciated by those skilled in the art, from a system perspective, the GUI could maintain the process bar 310 by adding an order to the list each time the trader submits a trade. The GUI could also monitor related instrument books to determine the status of respective orders. Alternatively, the GUI could monitor and update the status indicator from notifications received from the system 100 (e.g., notifications generated and received from the Book Manager 112c, etc.).

Alternative Settlement Dates

A feature of system 100 is that the Book Manager 112c maintains a separate book for each different settlement date for a given instrument. This permits traders to choose which date for the instrument to submit bids and offers for, as discussed above.

Broker Tools

System 100 preferably includes the capability for a third party, such as a brokerage firm, to monitor the market activity of traders on an approximate real-time basis, and also to act on behalf of certain traders in conducting market-activity.

These activities are performed in system 100 by use of "broker tools," software that is installed on desk-top workstations that are a part of system 100. These broker tools produce a GUI on the broker workstations that broker personnel can interact with. For example, a broker firm can be provided with the components depicted at client site 170 in Figure 1, except workstations 175 would be provided with the broker tools software. Only workstations equipped with the broker tools software are able to perform the functions unique to the broker tools, such as acting on behalf of a trader. Since workstations at the sites that are clients (e.g., traders) of the brokers, such as workstations 154, 160, 175 and 185 shown in Figure 1, will not typically be provided with the broker tools software, those workstations will not be able to perform the activities unique to the brokers tools software.

The function and operation of the broker tools software is described below with reference to Figures 41 - 50, which depict broker tools GUI screen displays. The preparation of the broker tools software would be readily apparent to a person of ordinary skill in the art, in view of this disclosure, preferably using Java beans and the C++ programming language. The broker tools software runs directly on top of the workstation's operating system. Given that the broker tools software is preferably Java-based, the broker tools software is platform-independent and will run on operating systems such as Windows 98, other Windows operating systems, UNIX, etc.

The broker tools software is launched by the user from his desktop workstation. Since the typical user will be a broker, the balance of this description of the broker tools software refers to the user as a "broker." After the broker tools software application is launched, a log-on screen (not shown) requests the broker to enter a user name and a password. Thereafter a broker desk tool bar having a plurality of icons is displayed on the workstation, as shown in Figure 41. This toolbar is used to launch particular broker tools applications by selecting the icon associated with each application, such as by mouse-clicking on the icon in the well-known manner.

In particular, clicking icon 10 of the broker desk tool bar launches the real time market monitor application, which displays the window shown in Figure 42. This window displays a scrolling list of the latest trading activity. Each row of information is for a different activity, and sets forth the time of the activity 10a, the instrument involved 10b, the type of activity 10c (whether it is a bid or offer for an instrument, or whether an instrument has been bought or sold), the price 10d, size of the order of transaction 10e;, the amount of any reserve 10f, the status 10g, the identity of the client conducting the activity 10h, and settlement convention 10i (e.g., a trade date "T" + 1 day, etc.).

To allow the broker tools to monitor only particular aspects of the market, such as the activities of particular clients, the broker creates filters by clicking on the "File" option 11, on the toolbar of the market monitor window, and selecting the "new filter" option" (not shown). This selection brings up a "create market monitor filter screen, shown in Figure 43, which contains three tabs. The first, the "orders/trades" tab 12, permits the broker to check the appropriate box so that the market monitor screen (Figure 42) displays only bids (12a), offers (12b), purchases (12c) or sales (12d), in accordance with the broker's selection. In addition, the broker can limit the display to activities of a certain size (12e), as well as apply further filtering criteria, depending upon the initial selections.

In addition to the "orders/trades" tab 12, the broker can also filter market activity based upon the instrument type. This is accomplished by clicking the "instruments" tab 13. Doing so launches a screen as shown in Figure 44. The broker then clicks on the "sectors" bar 13a, which reveals a drop-down menu 13b, shown in Figure 45. This drop-down menu contains a hierarchy of instrument groupings, as shown in the example set forth in Figure 45; for example, under "Instinet," there is "Fixed Income," which is further divided into "US Markets," "US Government," and then a number of U.S.-issues, such as "UST Notes/Bonds." If this latter group is selected, then a list of instruments 13c is displayed, as shown in Figure 46, and the user can then select some or all of these instruments by clicking on the appropriate buttons 13d, thereby monitoring activities in the selected instruments.

In addition to the filtering made available by selection of the "orders/trades" tab and the "instruments" tab shown in Figure 43, further filtering is also available by selecting the "clients" tab 14. Selecting this tab causes the broker tools GUI to display the window shown in Figure 47, which provides a list of clients 14a. Selecting a particular client will then cause that client's trader personnel to be displayed, by user ID and user name. Particular clients and/or traders can then be selected by clicking on the appropriate buttons 14b, thereby resulting in the activities of only the selected clients and/or traders to be displayed in the market monitor window shown in Figure 42.

To the right of the market monitor button 10 in Figure B1 is user monitor button 20. Activating this button causes the broker tools application to display a list of trades, by user ID and user name (Figure 48). In the case where the broker tools are being used by a broker, the broker tools software is configured to display only those traders associated with the broker: i.e., his clients. Upon selection of a particular trader, the screen shown in Figure 49 is displayed, which is a market level view similar to that shown in Figure 24, except that the order book will show the identity of the customers behind the bids and offers (21 and 22), and will also show any reserve behind each bid and offer (indicated parenthetically in the "size" column). In addition, this view allows the broker to act on behalf of the trader; thus, if the broker submits a bid or an offer, or hits or takes an order, that action will be registered as being undertaken on behalf of the selected trader. At the same time, the trader can continue to trade on his own behalf at his own workstation; utilization by the broker of the broker tools, to trade on the trader's behalf, does not disable the trader.

In addition to being able to conduct transactions on the trader's behalf, the broker tools software allows the broker to modify, cancel or suspend the trader's orders, whether made by the trader, or on behalf of the trader by the broker. To do this, the broker can either select a specific order, right-click on it, and a menu will appear allowing the user to select canceling, suspending or modifying the order. Alternatively, all orders for that trader can be suspended or canceled, either by actuating the appropriate button 23, 24, on the button bar shown in Figure 49, or by right-clicking on the selected trader in the list of traders, which as shown in Figure 50 allows the broker to cancel or suspend the orders for that selected trader.

The broker tools also permit the broker to view the selected client's blotter. This is done by clicking on the blotter icon 25, shown in Figure 49, or by selecting a trader in the view shown in Figure 48, and then clicking on the "view" label on the toolbar, which will reveal a "Blotter" drop-down selection. The broker can perform any activity in the blotter that is available to the trader, and the broker's activity will be reflected in the trader's own blotter.

The broker tools also allows the user to perform maintenance functions. In particular, clicking on the instruments icon 30, on the toolbar shown in Figure 41, bring up displays (not shown) allowing the broker to add new instruments to the system 100, as well as modify the characteristics of existing instruments. Also, clicking on icon 40 allows the broker to preform book maintenance, and clicking on the icon 50 allows the broker to preform sector maintenance. Book maintenance allows a broker through a dialog box, or a series of dialog boxes, to created "books" for instruments within the system 100. Book maintenance also allows a broker to open, close or deactivate a book. Sector maintenance is for maintaining sectors and creating new sectors. Sectors are broad categories for individual instruments. For example, a sector of U.S. Treasury Notes/Bonds could be 0-2 year, 2-5 year, 5-10 years, and 10-30 year sectors.

Supporting Network

Figure 15 illustrates a simplified view of the network supporting the system 100. In a preferred embodiment, the following networks are used to support the fixed income trading system: ISN/PIN; admin LANs in the data centers 110 and 120; ESN 114; and supporting networks such as 114a. The ESN, ISN and PIN networks are preferably TCP/IP based networks. As will be appreciated by those skilled in the art, any TCP/IP based network could be used by the system. Also, as will be appreciated by those skilled in the art, the network could include various routers, relays and switches to facilitate message distribution and system integrity. The PIN supports the system 100 by linking the data center 110 and data center 120 B-Desk trader desktop applications. With such a connection, a trader at a desktop application in location 110 can view the local network, as well as the network in the location associated with the data center 120. B-desk tool administrative functions can be exercised from a broker desk in various locations (e.g., in New York or London) with the preferred network connection. In a preferred embodiment, the PIN provides connection to the Client Database 135 and the

Terms and Conditions Database 136 by data centers 110 and 120, and facilitates message traffic between each server in a given data center. For example, in the Figure 2b embodiment, the PIN facilitates communication between server 110a and 110b. With this type of interconnectivity, all system activity can be accessed and processed through each B-Desk location. The PIN also provides links to the ESN network including for the broker desk, for example, to use the B-Desk tools.

As will be appreciated by those skilled in the art, known network protocols ensure that traffic automatically finds an alternative route in the event of a single router/switch or line failure. With the preferred connection discussed above, for example, if either B-desk is lost (e.g., in the event of a loss of a facility), the other B-desk will continue to have connectability to both data centers and clients via the esn network. Furthermore if either data center is lost both B-Desk locations have connectability to the remaining data center. The resiliency design of the system 100 provides support to ensure appropriate system response to a network failure. For example, if a router or switch that was associated with an applications server failed, the system 100 would utilize the remaining application server. The PIN can be monitored by operational personnel using software such as Hewlett Packard's Open View and Network Node Manager, for example, or other similar software platforms.

As will be appreciated by those skilled in the art, the admin LAN could support console access to each of the servers at each data center 110 and 120. Through the admin LAN, a system administrator could configure the network servers, administer the various domains, and perform other administrative tasks. In one preferred embodiment, each backup gateway 130 and 140 is connected to its respective data center through a LAN.

The ESN network 114 supports the fixed income trading system shown in Figure 15 by linking the components at data center 110 and 120 with various sites. In remote locations or in either of the locations associated with data center 110 and 120 the ESN network 114 can be fed (e.g., channeled) through existing or alternate networks such as network 114a (e.g., HPSN-NG, Reuter's High Performance Network-Next Generation). Like the PIN network, the ESN network 114 has numerous connection possibilities via the system switches and routers. As will be appreciated by those skilled in the art, standard network protocols ensure that traffic automatically finds an alternate route in the event of a single router, switch, or line failure. Like the PIN network the ESN network 114 can be monitored by operational personnel using software such as Hewlett Packard's Open View and Network Node Manager, for example, or any other similar software platform. The resiliency design of the system 100 provides support to ensure appropriate system response to a network failure. For example, if a router or switch that was associated with an application server failed, the fixed income trading system seamlessly utilizes the remaining application server.

U.S. Clearing and Settlement

Settlement for executed trades is facilitated in the United States through external agencies. Some of these agencies include ICI, Government Securities Clearing Corporation (GSCC), and the Chase Manhattan Bank. ICI is owned by Automated Data Processing, Inc., and handles post trading processing for settlement purposes. The ICI processing includes calculations such as end of money, accrued interest, and commission calculation.

ICI is a clearing entity 137 that the fixed income trading system can use. As discussed, a clearing entity such as ICI receives fixed income transaction information in real-time via a direct link with the system 100. The Back Office Adapter 112k is responsible for facilitating this data transfer. The Back Office Adapter 112k listens for tickets that have been produced by the Trade Manager 112d. Output from ICI may be forwarded to the Government Securities Clearing Corporation (GSCC) for further processing. Details regarding the ICI processing are provided below.

GSCC is a corporation owned by a variety of major U.S. security firms. GSCC was formed to overcome the inefficiency of manual clearing processes and to mitigate the risks associated with a failure of a securities firm. For example, if Firm A failed, all other firms with which Firm A had traded with would be unable to collect the money for trades they had assumed were final. This type of failure could result in other firms being unable to settle their trades, possibly resulting in other firm's failure as well.

GSCC mitigates this risk by becoming the counterparty to all transactions submitted to its netting service as explained below. Clients of GSCC are called participants. These clients must meet a variety of financial criteria in order to become participants. GSCC offers two essential services. First, GSCC offers a "trade comparison." Trade comparison matches trade details of a two-sided trade to confirm that a trade has indeed taken place. When GSCC clients produce a trade, each side submits its trade details to GSCC electronically. The comparison process runs each night. If a given side matches that of another, the trade is the to have been compared, and GSCC produces reports for each client showing the comparison activity. Most GSCC clients also utilize the "netting" services, explained below.

Netting is a process that calculates a firm's net position in each security trade for a given day. Netting is offered only to clients that have submitted their trades for trade comparison. For illustrative purposes, consider Firm A, which trades only one instrument all day. If Firm A buys and sells with numerous other firms, at day's end, Firm A and all other firms with which it traded submit their sides to GSCC for comparison. Once the comparison process confirms that all trades have actually occurred, the netting process calculates Firm A's net position in the security it traded. It does this by adding up all of the buys and all of the sells then calculating whether it has a positive (e.g., it will receive securities) or negative (e.g., it owes securities). As will be appreciated, this example is overly simplistic in that a firm is not likely to trade in only one security on a given day. However, the general process is the same regardless of how many securities a particular firm trades.

Part of the netting service is that GSCC becomes each submitting firm's counterparty for settlement purposes. This means that once two sides of a trade are to be compared, GSCC steps in and assumes the contraside for each submitting side for settlement. For example, Firm A and Firm B make a trade. Without GSCC, each firm would be exposed to the risk of the other firm failing to meet its settlement obligation. By having GSCC assume the contraside for settlement purposes, this risk is mitigated because it is highly unlikely that GSCC will default on any settlement obligation. The reason that this is true is that GSCC calculates a margin for each participating firm based on its trading activity. This margin is calculated and collected daily. Therefore, if any firm proves unable to meet its settlement obligation to GSCC, most of that obligation would already be deposited with GSCC. Any the amount not deposited with GSCC would be divided among all of the GSCC clients. GSCC mitigates the systematic risk through diversification of settlement failure.

Like comparison, netting is run in a batch mode in the early hours of the morning. When complete, reports are produced for submitting firms and net positions are reported to the clearing bank for settlement.

When a trade occurs through the system 100, as described previously, the system 100 steps in to protect each sides' identity by reporting itself as the contraside. For example, a case where Firm A and Firm B trade on the fixed income system is illustrated in Figure 16. Because the fixed income trading system 100 protects the identity of all trading clients, the system itself is considered the contraside from both Firm A's and Firm B's perspective. To settle a trade, Firm A and Firm B will report the trade to a clearing entity such as GSCC. Each submission will show the trading system 100 as the contraside. The system 100, through the Back Office Adapter 112k, will also report the trade to the clearing entity such as GSCC, via a clearing entity such as ICI, for example. The system 100 will show (via the Trade Manager 112d generating tickets as such) one submission with a trade with Firm A, and a second submission will show a trade with Firm B. As shown in Figure 17a, the system 100 posts (via the Back Office Adapter 112k) to the clearing entity ICI 137A a trade between itself and Firm A, and another trade between itself and Firm B. ICI or a similar clearing entity 137 then posts the transaction to a further clearing entity such as GSCC. Firm A and Firm B each posts a trade between itself and the system 100. In this manner the trades are matched up while still protecting the identity of each firm. A clearing entity 137 such as GSCC runs a comparison process similar to the one illustrated with Figure 17b. As seen in Figure 17b, the system 100 issues a submission (1) showing a trade with the system buying from Firm A and a submission (2) with the system selling to Firm B. As discussed, the submission could be a ticket created by the Trade Manager 112d, and forwarded by the Back Office Adapter 112k.

This information is submitted to a trading entity such as GSCC in the process described above with respect to Figure 17a. Firm A also issues a submission (1) indicating a sell from itself to the system 100. Similarly Firm B issues a submission (1) indicating a buy from itself from the system 100. The GSCC then compares the system's 100 buy submission (1) with Firm A's sell submission (1). The system's sell submission (2) is compared with the Firm's B buy submission (2).

When GSCC runs the netting process, the following results occur. First, the system 100 is "flat" because the system's buy/sell submissions cancel each other out. Firm A will "net long," since it owed money and must surrender the security. Firm B will "net short," since it must pay money to receive the security B. The next step involves an actual settlement, which is facilitated by a settlement organization such as Chase bank, as discussed below.

The Chase bank provides the financial services necessary to settle transactions. In one example, the GSCC reports the netting results to Chase, which instructs Chase as to the day's transaction details. GSCC becomes the contraside to all transaction for settlement purposes. Under the example, GSCC would then tell Chase to expect to receive securities from Firm A and to receive payment from Firm B. Upon receipt, Chase transfers the money to Firm A's account and the securities to Firm B's account.

Non-U.S. Clearing and Settlement

The following discussion details an embodiment where data center 120 is located in a European market. Typically, settlement is facilitated in Europe through external agencies, such as CSDs (Central Securities Depositories); ICSDs (International Securities Depositories); and Settlement Agents.

The International Securities Market Association (ISMA) is a self-regulatory industry body and trade association for the international securities market in Europe. ISMA's primary role is to issue rules and recommendations relating to trading and settlement practices in the international market. ISMA membership currently includes more than 700 member organizations from fifty countries. One of the ISMA's rules is that all European trades be reported to ISMA within 30 minutes. To facilitate this requirement, an ISMA system called "TRAX" is used.

TRAX is ISMA's real-time electronic trade matching, confirmation and regulatory reporting system. All executions on the system 100 must be reported to TRAX by the system 100 and the two client contra-sides. TRAX compares all sides of an execution and reports results back to all submitting sides. Confirmation that all submitting sides are in agreement regarding execution details does not guarantee that a given trade will settle. The purpose of TRAX reporting is to alert submitters to errors that could interfere with settlement in a timely manner.

A Central Securities Depository (CSD) is an institution that provides central vaulting of certificates (if certificates exist) and maintains security transfer and ownership records. CSDs provide domestic settlement services, meaning that they serve clients domiciled in a given country. Specific services can vary from one country to another, but they generally act as Transfer Agent and Paying Agent. If a customer of a CSD needs to settle a cross-border trade, the local CSD can interact with an ICSD or a Settlement Agent to facilitate the settlement process.

An International Central Securities Depository (ICSD) is similar to a CSD, except that its services are used to settle "cross-border" trades. An ICSD acts as Transfer Agent and Paying Agent. CSDs can interact with ICSDs in order to facilitate settlement of cross-border trades. From a system perspective, the system deals with ICSD which reports to an agent. The agent typically uses a local settlement processes.

Two examples of ICSDs are Euroclear and Cedel. Euroclear is an international depository, located in Brussels, formed to provide exchange and clearance services for internationally traded securities. Euroclear has an "electronic bridge," or a direct link, to the Cedel depository to facilitate settlements between the two ICSDs. The system 100 can maintain accounts with an ICSD for settlement purposes.

Cedel (Centrale de Livraison de Valeurs Mobilieres) is the second major depository, along with Euroclear, for international trading. Located in Luxembourg, Cedel works with Euroclear to facilitate transaction flows. An example of the cooperation between these two depositories, which results in efficiency for participants, is that Cedel performs pre-matching of trades with Euroclear daily to ensure timely delivery and receipt. Benefits of Cedel's working relationship with Euroclear include fewer physical settlements which result in lower fees charged to participants, and same day cache proceeds.

Settlement Agents serve as a local intermediary between the system 100 (via its Euroclear account) and local CSDs. Settlement Agents facilitate efficiency. For example, in order to do business with a given CSD, it is necessary to meet specific business and regulatory requirements. Because of the diversity of requirements among CSDs and the differing regulatory environments of each country, it is efficient to use a Settlement Agent that meets all the necessary requirements to do business in a local market. Preferably, the system uses two such agents: Citibank and Paribus.

The following discussion relates to systems and components that facilitate settlement in a European marketplace. A "Capital Markets Package" (CMP) 240 is software licensed through Wilco International Limited. CMP 240 is the system's European "books and records" and the main "engine" driving a European settlement model that requires minimal manual intervention. The CMP 240 interfaces with the core components 122 via the Back Office adapter 122k, and is preferably resident in a data center (e.g., 120) corresponding to the European market, for example. From a functional viewpoint, the Back Office Adapter 122k preforms in a similar manner as does its U.S. market counterpart.

When the CMP 240 receives a trade, it enhances the trade data with settlement information from the static data stored in its own database. This information is important because there is no single entity responsible for settlement. Because the system 100 acts as the contra-side for each trade (thereby ensuring the anonymity of each trade side), the system 100 must settle with each client according to each client's preference. For example, one client may utilize a local CSD, while another may have an account with Cedel. CMP 240 inserts settlement information of this kind so the correct settlement instructions are given to the various software components that facilitate settlement of the fixed instrument transactions.

The CMP 240 generates a TRAX instruction to meet the TRAX reporting requirement. CMP 240 also receives messages from the TRAX system that advise CMP 240 of the trade's status (e.g., whether its details matched those reported by the contra-side). The CMP 240 can also generates a fax confirmation to the counterparts, if required by a client. The CMP 240 generates a SWIFT settlement message to instruct the settlement entities regarding the trade and facilitate the settlement transaction. CMP also receives SWIFT messages, as discussed below, to update the settlement status of transactions. The CMP 240 also issues instructions to SmartStream, the auto-reconciliation software, discussed below, produces files for system 100 finance and compliance departments for tracking purposes, and archives the transaction to an optical disk (not shown).

As will be appreciated by those familiar with fixed income instruments, SmartStream is a reconciliation system for Depots and Nostro accounts, trade accounts and for transactions. SmartStream provides information regarding the settlement status of trades, as well as reconciliations among the various systems and clearing entities.

SWIFT is an acronym for Society For Worldwide Interbank Financial Telecommunications. SWIFT is a system of electronic communication that facilitates the exchange of trading and settlement messages, as well as payment instructions and cache transfers between financial institutions around the world. SWIFT messages can be sent both internationally and domestically.

SWIFT provides common rules, standards and communication methods for users of financial services. SWIFT protects against unauthorized access, loss or incorrect delivery of messages, transmission errors, loss of confidentiality and fraudulent changes to messages.

The following discussion describes how a trade is processed by the system's components and external settlement agencies to facilitate settlement in a European market.

When a trade occurs, the system 100 protects each side's identity by reporting itself as the contraside. For example, consider a situation where Firm A and Firm B trade on the system 100, as illustrated in Figure 19. The system 100 protects the identity of all clients, so the system 100 is the contra-side from Firm A and Firm B's perspective. This execution information is provided to CMP 240 via the Back Office Adapter 122k for post-trade processing.

Reporting to TRAX must occur within thirty minutes of execution. When CMP 240 receives an execution from the Back office Adapter 122k, it immediately issues a message to the TRAX server 251, which forwards the information to the TRAX system 250. SmartStream 260 also receives notification that the TRAX message was sent, so it can monitor the response from TRAX. Each contra-side also reports the trade to TRAX within the thirty minutes, as shown in Figure 20.

CMP 240 enriches the execution data with settlement instructions, and issues a SWIFT message to the entities involved in the settlement. SmartStream 260 also receives the SWIFT instructions and updates the trade status accordingly. Typically, the CMP 240 enriches the execution data with settlement instructions and issues a SWIFT message immediately after it sends trade notification to the TRAX system 250. The

CMP 240 does not wait for a response back from TRAX before proceeding with post trade processing.

Settlement information for each system 100 client is maintained directly in the CMP 240, using a menu-driven administrator interface 241. Settlement instructions are necessary because there is no single centralized clearing and settlement entity in Europe. The entity utilized to facilitate settlement (e.g., a ICSD or CDS) varies from location to location.

The TRAX system 250 matches the trade details and reports the results back to all sides, as shown in Figure 21. A TRAX-reported match is *not* a confirmation that the trade will settle. TRAX reporting only confirms whether the reported trade details match among the parties involved. CMP 240 and SmartStream 260 receives the TRAX response and updates the trade status accordingly. Notification of a failed trade can be issued through ISCD, via SWIFT.

Once settlement has occurred, the clearing entities 138 involved notify the system 100 and the contra-side firms via SWIFT. SmartStream 260 listens for the incoming SWIFT messages and updates the trade status accordingly (see Figure 22). At the end of the trading day, CMP 240 batch processes the day's activity and creates files to update the Finance and Compliance departments' systems (see Figure 22).

While the present invention has been described with respect to what is presently considered to be the preferred embodiments, it is to be understood that the invention is not limited to the disclosed embodiments. To the contrary, the invention is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims. The scope of the following claims is to be accorded the broadest interpretation so as to encompass all such modifications and equivalent structures and functions.

What is Claimed:

1. A trading system for trading fixed income instruments, the system comprising:

at least two server computers, each comprising:

a first interface to communicate with a server computer;

a second interface through which orders are received by said system;

at least one processor for executing computer code;

a first database pertaining to trading activity through said system in a first trading market;

a second database pertaining to trading activity through said system in a second trading market;

at least one cache storing information regarding market activity in the respective first or second market;

a memory having computer executable code stored thereon, the code for (i) updating said first and second databases and said at least one cache; (ii) synchronizing the updates; and (ii) processing orders from each respective trading market.

2. The trading system according to Claim 1, wherein each of said two server computers is located in a geographically separate location.

3. The trading system according to Claim 2, further comprising at least one trader site comprising at least one application server and a workstation connected to said at least one application server.

4. The trading system according to Claim 3, wherein said at least one application server contains a cache having at least a copy of the information in said first database stored thereon.

5. The trading system according to Claim 4, wherein the code i) updates the cache in the at least one application server, and ii) synchronizes the updates with the updates to said first and second databases.

6. The trading system according to Claim 1, wherein the code maintains at least one instrument book defined by a set of bids and offers and by a settlement date.

7. The trading system according to Claim 6, wherein the code changes a status of the at least one instrument book.

8. The trading system according to Claim 7, wherein a status comprises open, closed, inactive and retired.

9. The trading system according to Claim 6, wherein the code ranks a bid having a highest price first among other bids in the at least one book.

10. The trading system according to Claim 6, wherein the code ranks an offer having the lowest price first among other offers in the at least one book.

11. The trading system according to Claim 6, wherein the code ranks passive orders having a same price in the at least one instrument book on a first come, first serve basis with respect to each other.

12. The trading system according to Claim 6, wherein the code processes aggressive orders on a first come, first serve basis.

13. The trading system according to Claim 6, wherein the code removes from the at least one book a passive bid or passive offer once the passive bid or passive offer has been executed.

14. The trading system according to Claim 6, wherein the code validates orders.

15. The trading system according to Claim 6, wherein a user terminal is interfaced with the system.

16. The trading system according to Claim 15, wherein the at least one book is configured for display through the user terminal.

17. The trading system according to Claim 14, wherein the code filters data transmitted to the user terminal.

18. The trading system according to Claim 1, wherein the code causes said system to transfer information related to executed transactions through the trading system to at least one clearing entity.

19. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) processing a sweep order for the at least one instrument book.

20. The trading apparatus according to Claim 19, wherein the sweep order is an order to aggress multiple passive orders at a selected price option and amount.

21. The trading apparatus according to Claim 20, wherein the code aggresses passive orders to fill the sweep order by sequentially i) executing shown amounts of passive orders at the selected price; ii) executing reserve amounts at the selected price when needed to fill the sweep order; and iii) executing passive orders at a next price when needed to fill the sweep order.

22. The trading apparatus according to Claim 21, wherein the code expires a first passive order listed in the at least one instrument book when i) the code determines that the first passive order was submitted by a trader who also submitted the sweep order, and ii) the code executes a second passive order that is listed below the first passive order.

23. The trading apparatus according to Claim 19, wherein a first passive order comprises price and amount information.

24. The trading apparatus according to Claim 23, wherein the first passive order further comprises shown amount information, and the code determines a reserve amount from the shown amount.

25. The trading apparatus according to Claim 24, wherein at least one user terminal is interfaced with said trading apparatus and the code conditionally conceals the reserve amount from the at least one user terminal.

26. The trading apparatus according to Claim 25, wherein the codes conveys the reserve amount to the at least one user terminal when a condition comprises a user who entered the first passive order.

27. The trading apparatus according to Claim 24, wherein the code executes a shown amount prior to executing a reserve amount.

28. The trading apparatus according to Claim 27, wherein the code executes shown amounts for orders at a same price in an instrument book before executing reserve amounts for the orders at the same price.

29. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders,

wherein said steps process aggressive orders each comprising instrument identification, price option information, amount information and minimum amount for execution information.

30. The trading apparatus according to Claim 29, wherein the code executes a trade only when a total amount of shown and reserve amounts for orders at the order price option

in the at least one instrument book equals at least the minimum amount for execution.

31. The trading apparatus according to Claim 30, wherein the code generates an execution notification regarding the order only when a trade is executed.

32. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) wherein said steps process aggressive orders each comprising instrument identification, price option criteria, amount information, and a post residual authorization.

33. The trading apparatus according to Claim 32, wherein the code converts an unfilled balance of the first aggressive order into a passive order having an aggressed-on price.

34. The trading apparatus according to Claim 33, wherein the code posts the unfilled amount in the at least one book as a shown amount.

35. The trading apparatus according to Claim 33, wherein the code posts the unfilled amount in the at least one instrument book as a shown amount and the remainder as a reserve amount.

36. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) suspending all orders submitted by a user through a user terminal when communication with the user terminal is interrupted.

37. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) canceling all orders received from a user through a user terminal in response to a user generated bulk cancel request.

38. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) suspending all orders received from a user through a user terminal in response to a user generated bulk suspend request.

39. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) converting passive orders into an aggressive orders and executing the converted aggressive orders at a top of book price, when the passive orders invert a market.

40. The trading apparatus according to Claim 39, wherein a first passive order inverts the market when the first passive order comprises a bid having a higher price than the lowest priced offer in the at least one instrument book,

or when the first passive order comprises an offer having a lower price than the highest priced bid in the at least one instrument book.

41. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) expiring all orders at the end of a trading day.

42. A trading apparatus comprising:

at least one network server comprising:

at least one processor which executes computer executable code;

an interface through which orders are received by said trading apparatus;

a memory for storing computer executable code, the code including steps for (i) maintaining at least one instrument book defined by a set of bids and offers and by a settlement date; (ii) processing passive orders; (iii) processing aggressive orders; (iv) executing trades based on the aggressive orders; and (v) expiring orders based on user selected criteria.

43. The trading apparatus according to Claim 42, wherein the code expires a first order, comprising good until topped user selected criteria, when a second order having a

better price than the first order is entered into the at least one instrument book.

44. The trading apparatus according to Claim 42, wherein the user selected criteria comprises good until topped, good until time, good until canceled, and good until end of day.

45. Media containing computer executable code to generate a graphic user interface invocable by programable software for use in a fixed income trading system, the system providing a computerized venue for trading instruments, the system receiving orders including bids and offers for individual instruments submitted to the system, the system maintaining at least one book defined by a set of orders and by a settlement date, the graphic user interface for employment on a trader work station connected to said system, the computer executable code comprising:

steps to generate a first window comprising a first display region displaying a top of book view, a second display region displaying a key issues view, a third region displaying a depth of book view, and a forth region including at least one user activatible button to invoke an order entry dialog box through which a user inputs order requests into the system.

46. The media according to Claim 45, wherein the computer executable code further comprises steps for generating a region including a user activatible button for displaying a floating book window.

47. The media according to Claim 46, wherein the floating book comprises a depth of book view and a region including user activatible buttons to invoke at least one order entry dialog box.

48. The media according to Claim 45, wherein said first window further comprises a region having a pull-down menu including a user activated preference dialog box, the dialog box accepting user entered default settings for orders.

49. The media according to Claim 48, said computer executable code further includes steps for generating a shortcut order entry menu for invoking an order entry dialog box comprising the default settings for the individual instrument, when an individual instrument displayed on said first window is activated.

50. The media according to Claim 45, wherein said first window further comprises a fifth region for displaying a moving real-time ticker comprising instrument information.

51. Media containing computer executable code to generate a graphic user interface invocable by a software program, said graphic user interface for use by brokers in a trading system, the trading system providing a computerized venue for trading instruments, the code comprising:

steps to generate a first window displaying a scrolling list of real-time market activity that occurs through the system, the first window including a first region having a pull-down menu for a user activatible filter for selecting a subset of activity to view for real-time display; and

steps to generate a second window displaying a client-specific blotter comprising order history information, order status information, and user activated transactions,

wherein a broker accessing the second window can interface with the system on behalf of the specified client.

52. A graphic user interface according to Claim 51, wherein an activity subset comprises market type, instrument or client.

53. A network trading system comprising:

at least one server including at least one processor for executing computer code;

an interface through which a user gains access to said system;

at least one memory having computer executable code stored thereon, the code for: (i) tracking activity through the system with respect to individual trading instruments; (ii) organizing for real-time display the tracked activity of individual trading instruments; and (iii) displaying to the user a subset of the tracked activity.

54. A fixed income securities trading system, comprising:

a data hub including a processor and a memory for storing and updating a plurality of trade books, each book corresponding to at least one fixed income securities instrument, each book having data corresponding to (i) system clients, (ii) offers of said clients for sales of said at least one fixed income securities instrument, (iii) bids of said clients for purchases of said at least one fixed income securities instrument, and (iv) status of said offers and said bids; and

a plurality of trader workstations, each coupled to said data hub and displaying at least one of said plurality of trade books, each workstation transmitting to said data hub (i) offers of the said clients for sales of said at least one fixed income securities instrument, (ii) bids of said clients for purchases of said at least one fixed income securities instrument, and (iii) in instruction to either accept an offer for sale or a bid for purchase of said at least one fixed income securities instrument;

said data hub, in response to receipt of said instruction, executing the trade and updating said book status, said data hub outputting a clearing entity resolution signal to a clearing entity.

55. A method of operating a computerized fixed income securities trading system, comprising steps of:

storing in a computerized data hub a plurality of trade books, each book corresponding to at least one fixed income securities instrument, each book having data corresponding to (i) system clients, (ii) offers of said clients for sales of said at least one fixed income securities instrument, (iii) bids of said clients for purchases of said at least one fixed income securities instrument, and (iv) status of said offers and said bids;

displaying said each book at a plurality of remote computerized trader workstations coupled to said data hub;

updating said each book in response to a client offer for sale of said at least one fixed income securities instrument input from a trader workstation;

updating said each book in response to a client bid for purchase of said at least one fixed income securities instrument input from a trader workstation;

in response to in instruction from a trader workstation to either accept an offer for sale or a bid for purchase of said at least one fixed income securities instrument, executing the trade and updating said book status; and

after the execution of the trade, outputting a clearing entity resolution signal to a clearing entity.

FIG. 1

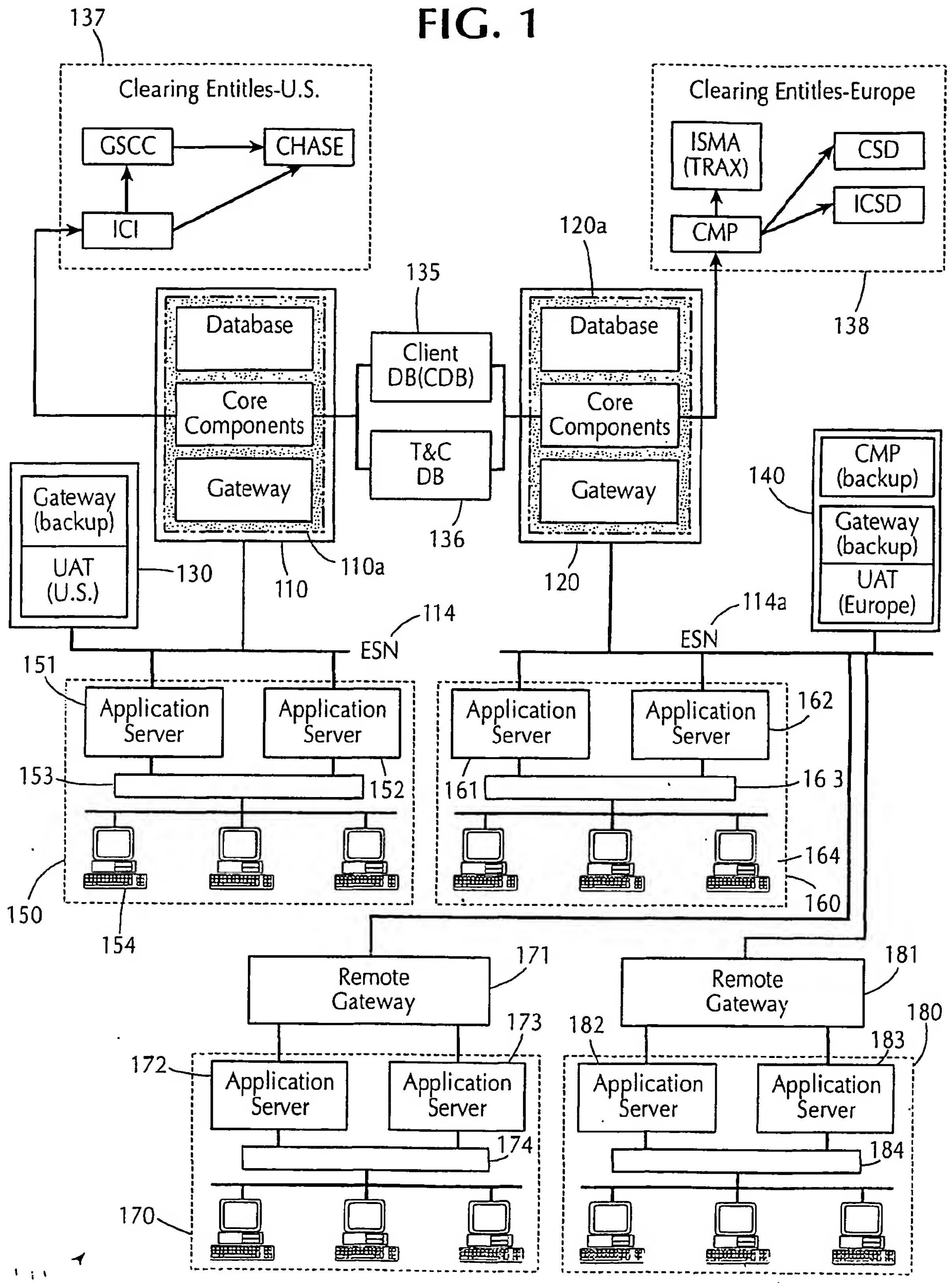
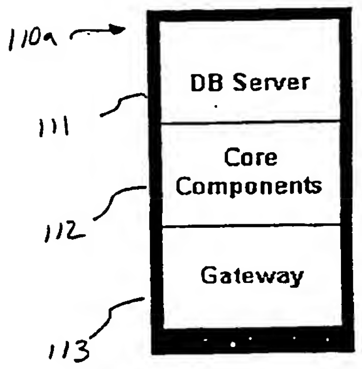

137
Clearing Entitles-U.S.
GSCC
CHASE
ICI
Clearing Entitles-Europe
ISMA (TRAX)
CSD
ICSD
CMP
120a
138
135
Database
Core Components
Gateway
Client DB(CDB)
T&C DB
136
Gateway (backup)
UAT (U.S.)
130
110
110a
120
140
CMP (backup)
Gateway (backup)
UAT (Europe)
114
ESN
114a
ESN
151
Application Server
Application Server
152
153
150
154
161
162
16 3
164
160
Remote Gateway
171
Remote Gateway
181
180
172
173
174
170
182
183
184

Figure 2a



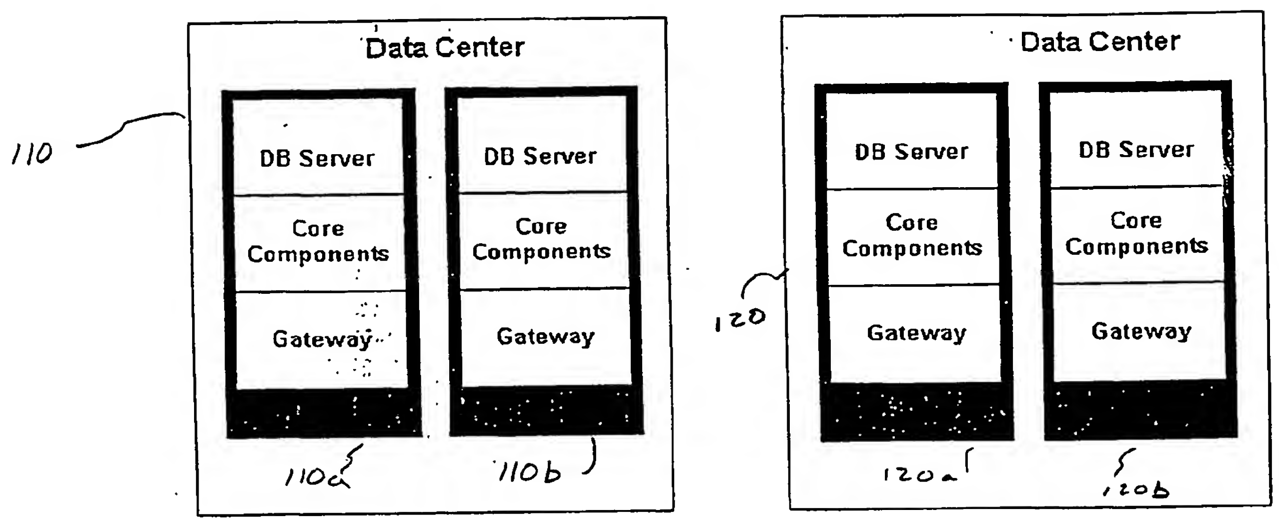


Figure 2b

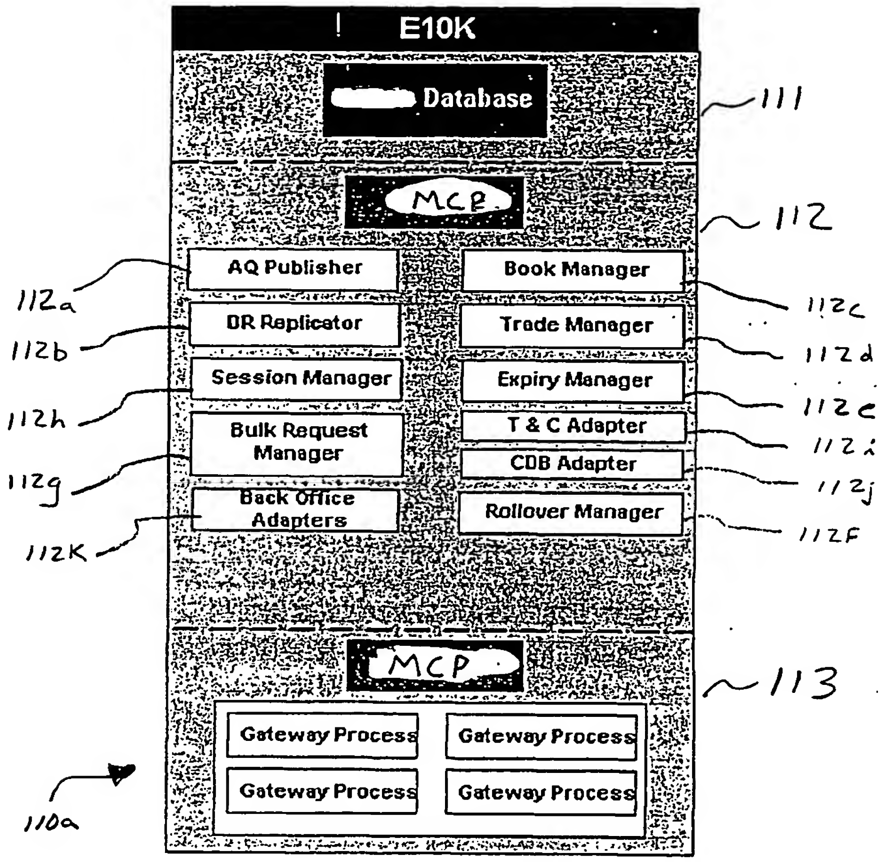

E10K
Database
MCP
AQ Publisher
Book Manager
DR Replicator
Trade Manager
Session Manager
Expiry Manager
Bulk Request Manager
T & C Adapter
CDB Adapter
Back Office Adapters
Rollover Manager
MCP
Gateway Process
Gateway Process
Gateway Process
Gateway Process
111
112
112a
112b
112h
112g
112K
112c
112d
112e
112i
112j
112F
113
110a

Figure 3

Figure 4

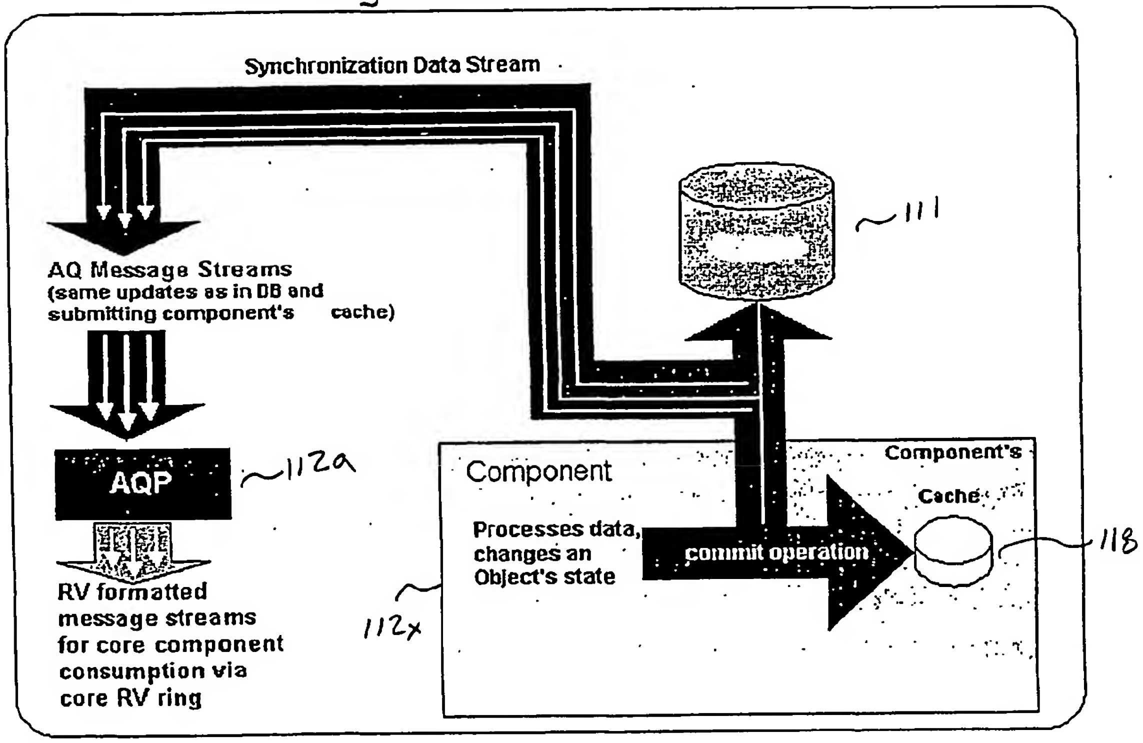
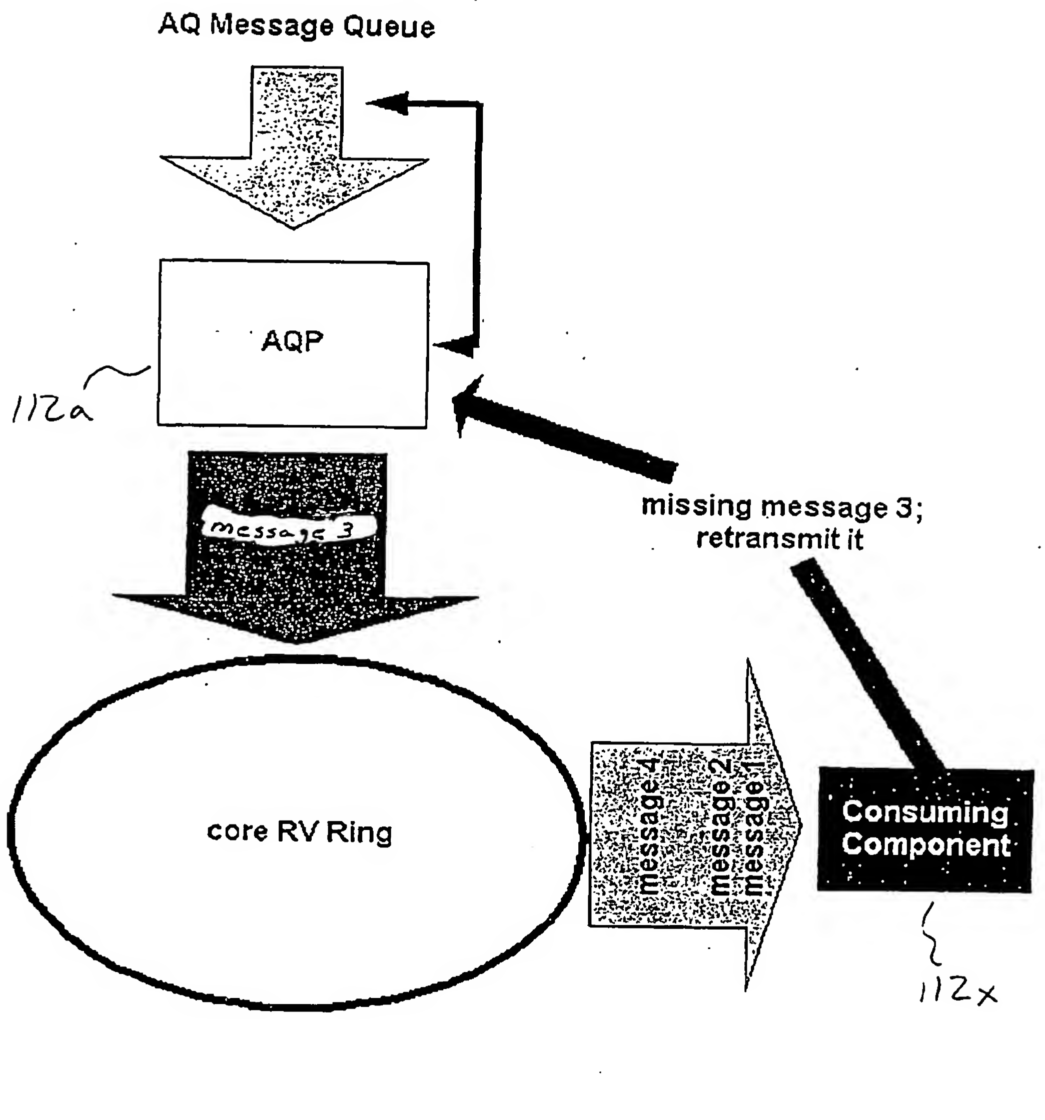

Synchronization Data Stream
AQ Message Streams
(same updates as in DB and submitting component's cache)
AQP
~112a
RV formatted message streams for core component consumption via core RV ring
~111
Component
Processes data, changes an Object's state
commit operation
Component's Cache
118
112x

Figure 5


AQ Message Queue
AQP
112a
message 3
missing message 3; retransmit it
core RV Ring
message4
message2
message1
Consuming Component
112x

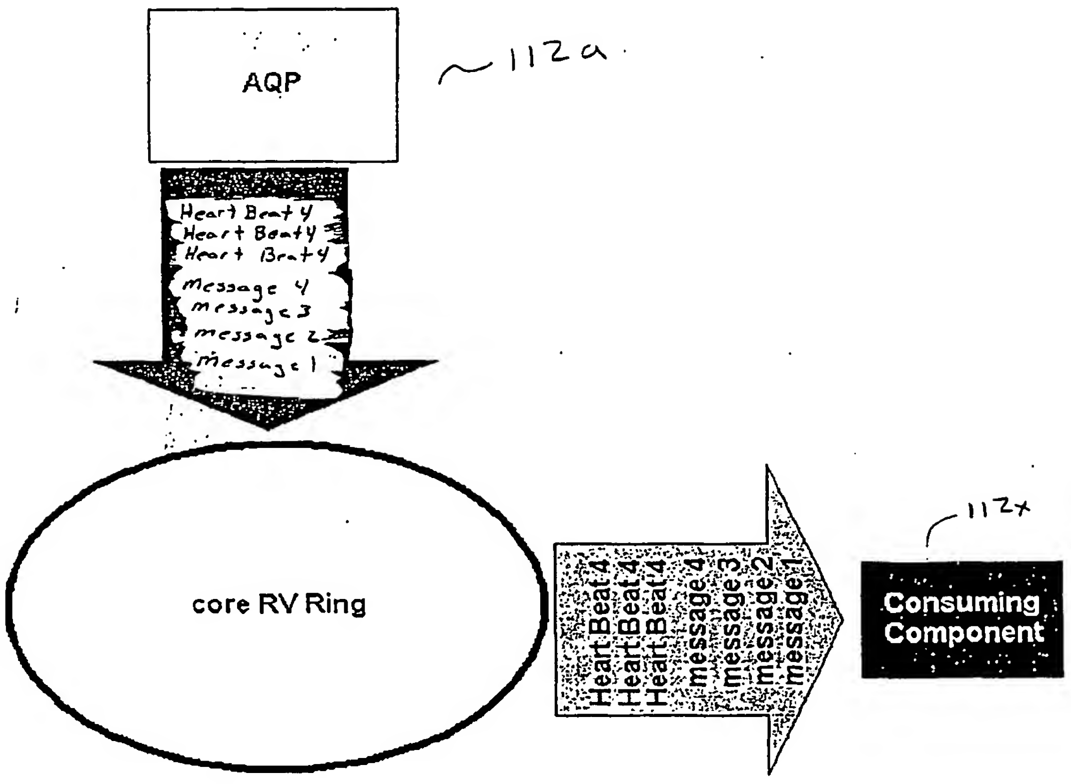

AQP
112a
Heart Beat 4
Heart Beat 4
Heart Beat 4
message 4
message 3
message 2
message 1
core RV Ring
Heart Beat 4
Heart Beat 4
Heart Beat 4
message 4
message 3
message 2
message 1
112x
Consuming Component

Figure 6

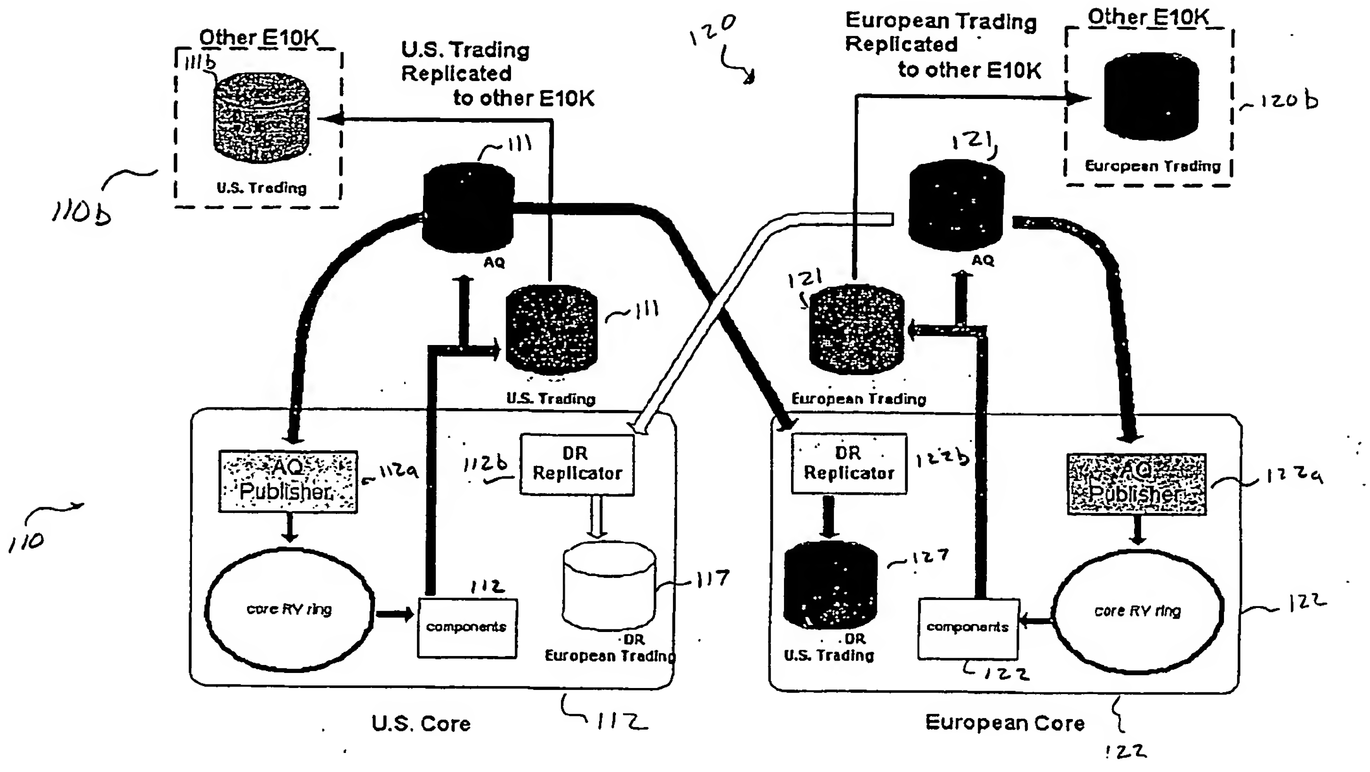

Other E10K
111b
U.S. Trading
110b
U.S. Trading
Replicated
to other E10K
111
AQ
111
U.S. Trading
120
European Trading
Replicated
to other E10K
Other E10K
120b
European Trading
121
AQ
121
European Trading
110
AQ Publisher
112a
112b
DR Replicator
core RV ring
112
components
117
DR
European Trading
U.S. Core
112
DR Replicator
122b
AQ Publisher
122a
127
DR
U.S. Trading
components
122
core RV ring
122
European Core
122

Figure 7

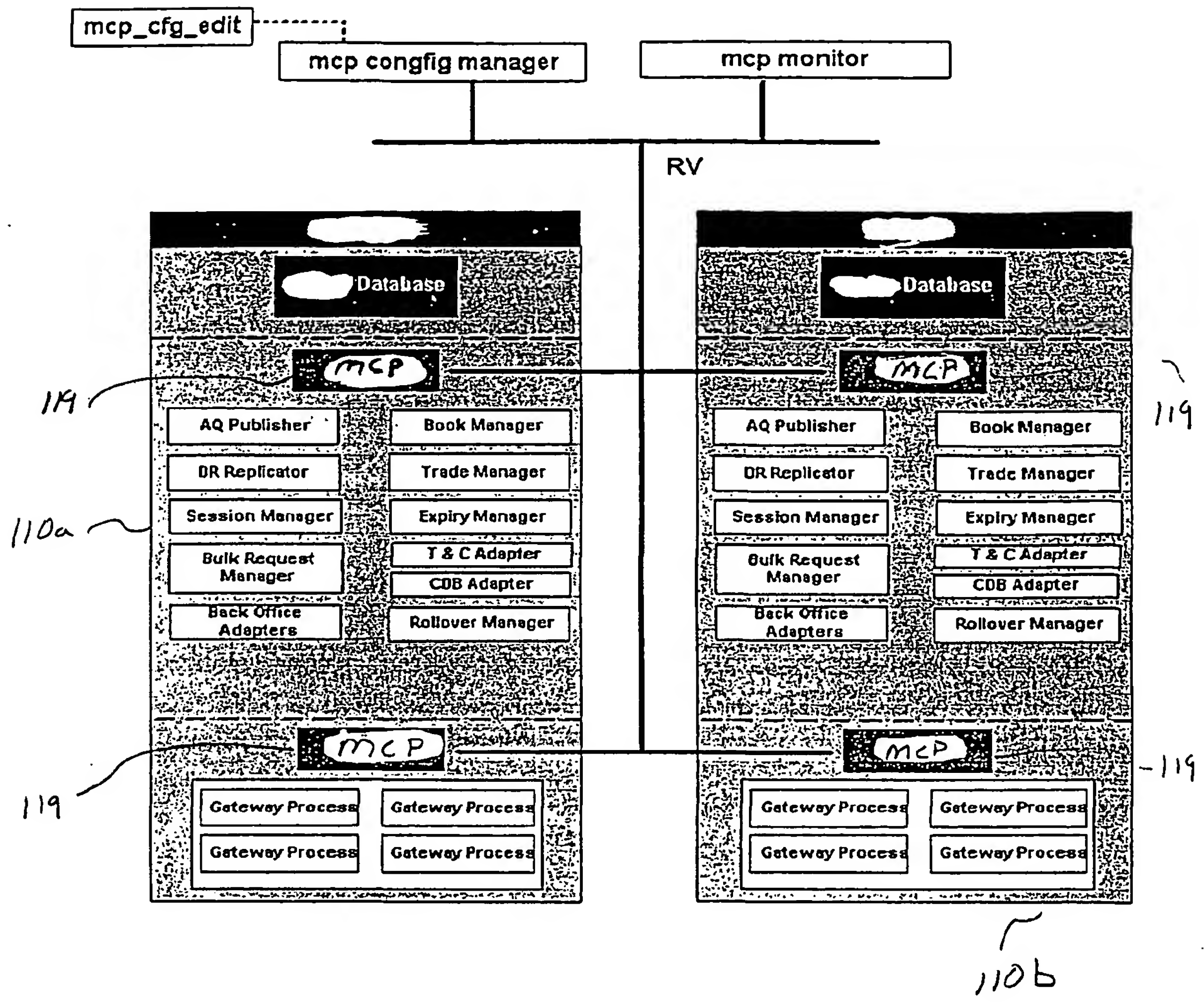
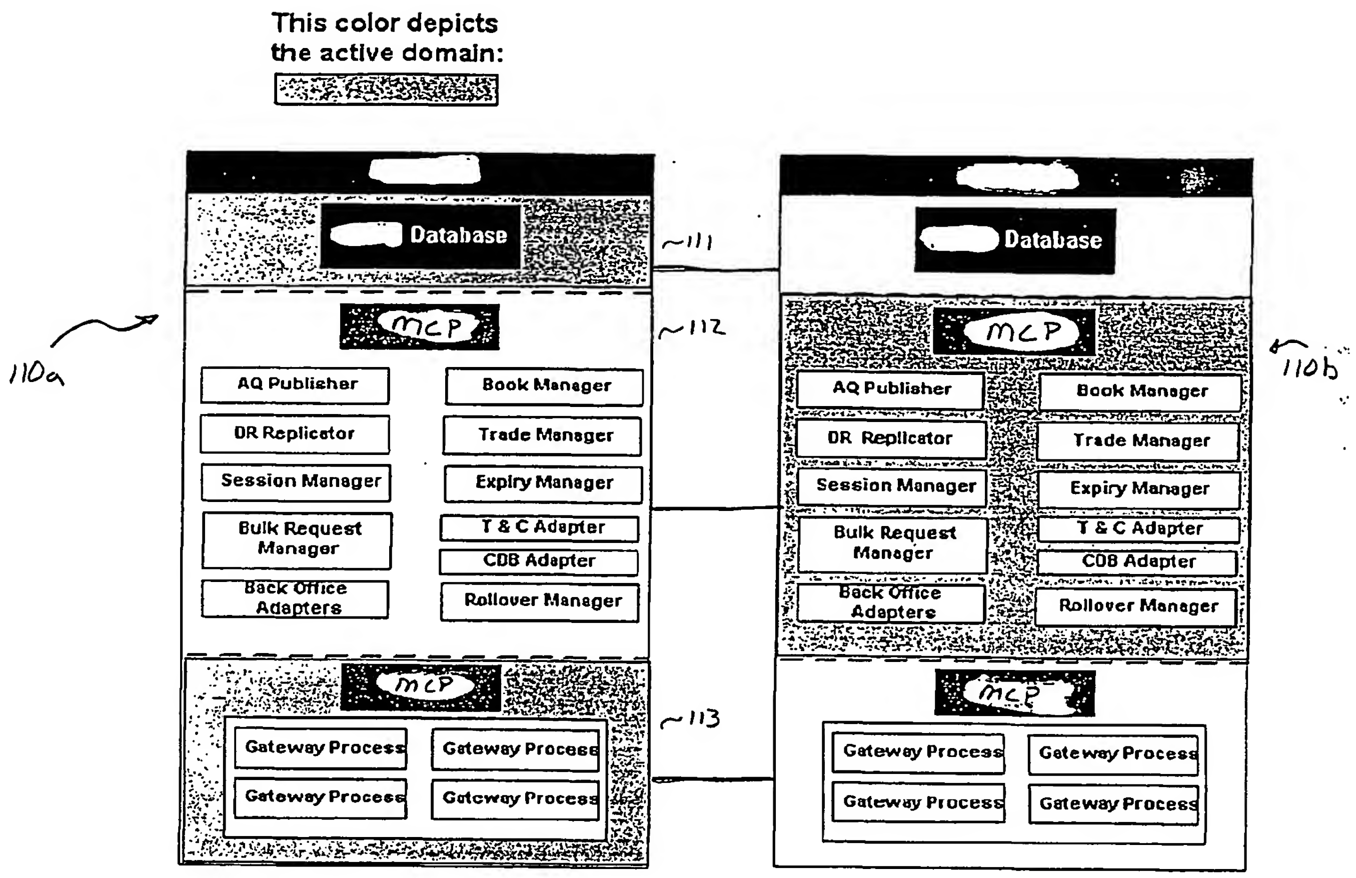

mcp_cfg_edit
mcp congfig manager
mcp monitor
RV
Database
Database
mcp
MCP
119
119
AQ Publisher
Book Manager
DR Replicator
Trade Manager
Session Manager
Expiry Manager
Bulk Request Manager
T & C Adapter
CDB Adapter
Back Office Adapters
Rollover Manager
110a
mcp
mcp
119
119
Gateway Process
110b

Figure 8



This color depicts the active domain:
Database
MCP
AQ Publisher
DR Replicator
Session Manager
Bulk Request Manager
Back Office Adapters
Book Manager
Trade Manager
Expiry Manager
T & C Adapter
CDB Adapter
Rollover Manager
MCP
Gateway Process
Gateway Process
Gateway Process
Gateway Process
~111
~112
~113
110a
110b
Database
MCP
AQ Publisher
DR Replicator
Session Manager
Bulk Request Manager
Back Office Adapters
Book Manager
Trade Manager
Expiry Manager
T & C Adapter
CDB Adapter
Rollover Manager
MCP
Gateway Process
Gateway Process
Gateway Process
Gateway Process

Figure 9

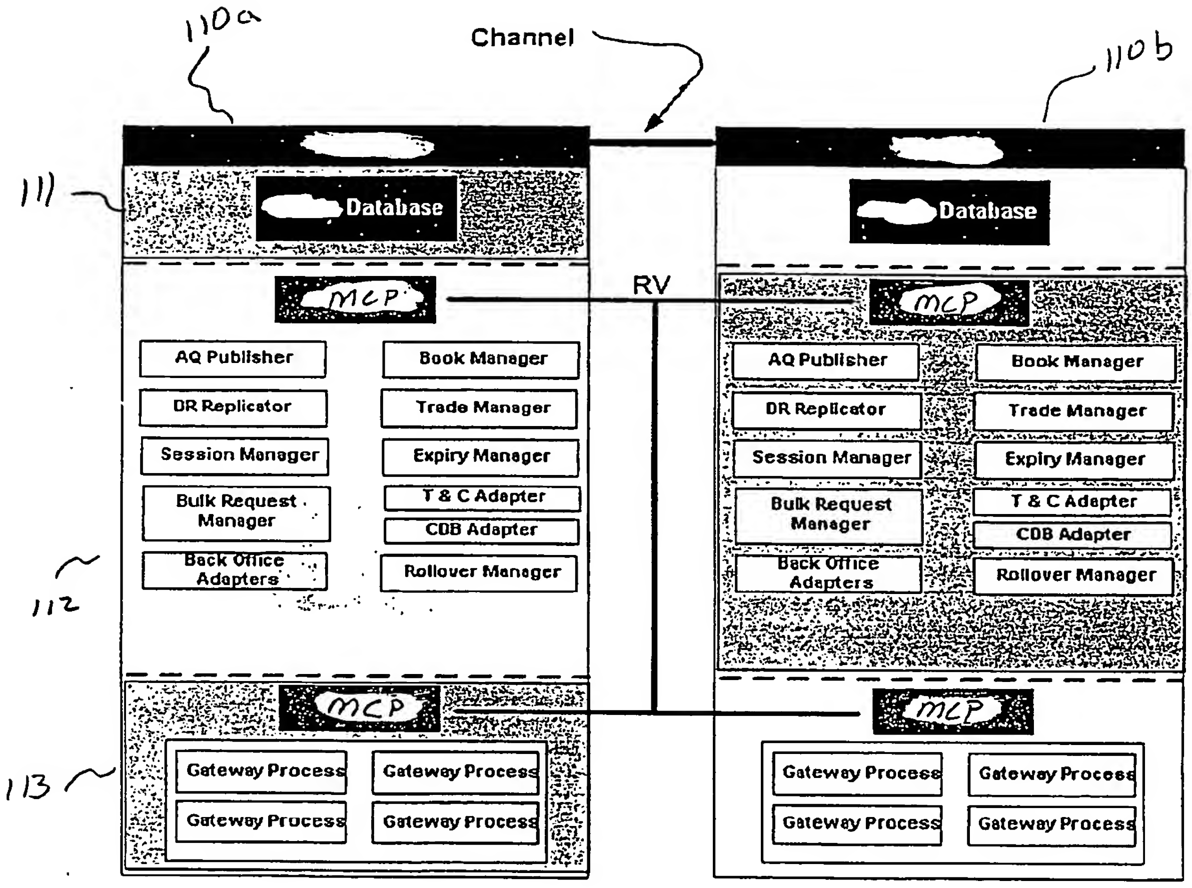

110a
Channel
110b
111
Database
Database
RV
MCP
MCP
AQ Publisher
Book Manager
DR Replicator
Trade Manager
Session Manager
Expiry Manager
Bulk Request Manager
T & C Adapter
CDB Adapter
Back Office Adapters
Rollover Manager
112
MCP
MCP
113
Gateway Process
Gateway Process
Gateway Process
Gateway Process

Figure 10

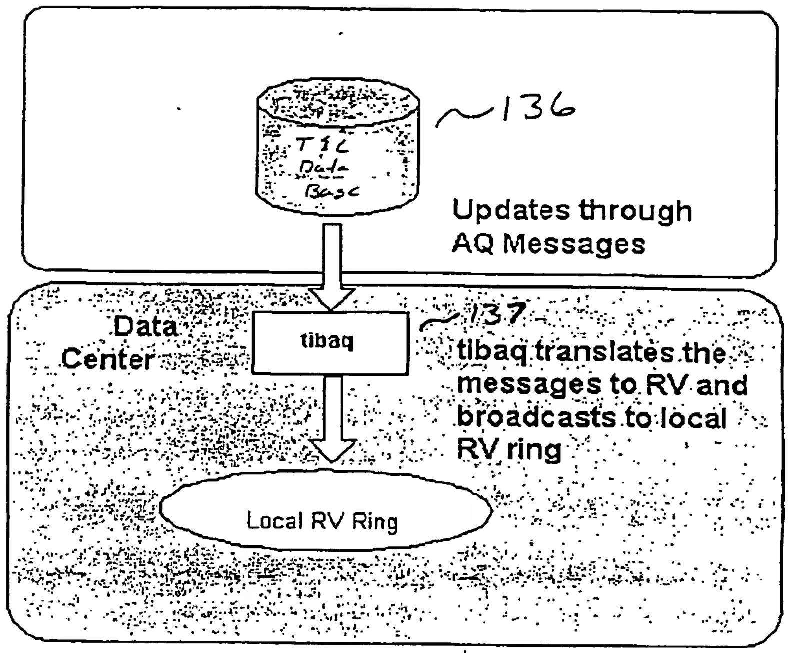

T&C Data Base
136
Updates through AQ Messages
Data Center
tibaq
137
tibaq translates the messages to RV and broadcasts to local RV ring
Local RV Ring

Figure 11a

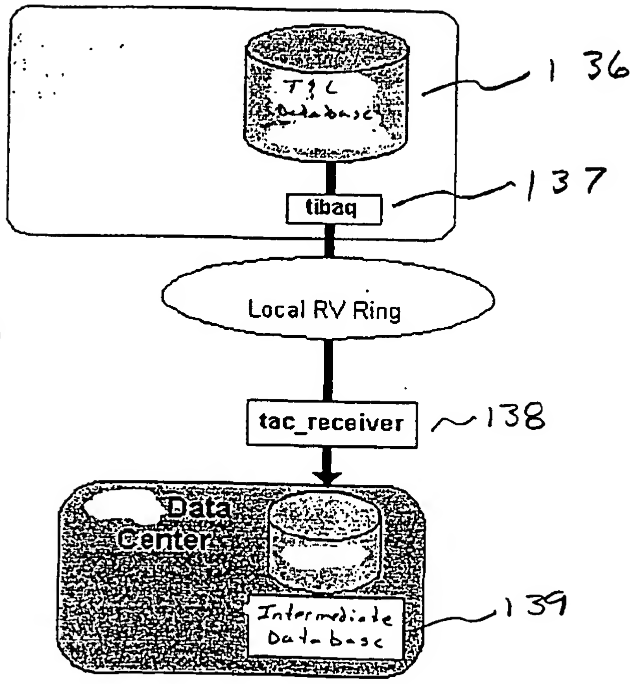


Figure 11b

Figure 11c

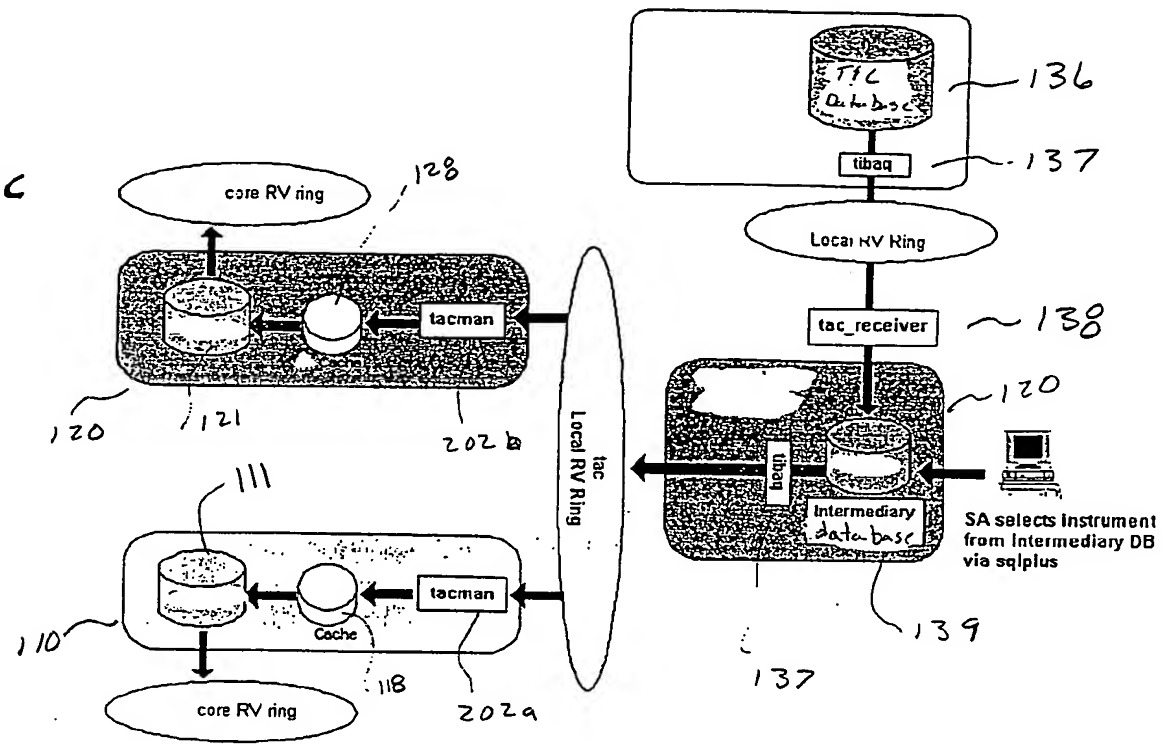
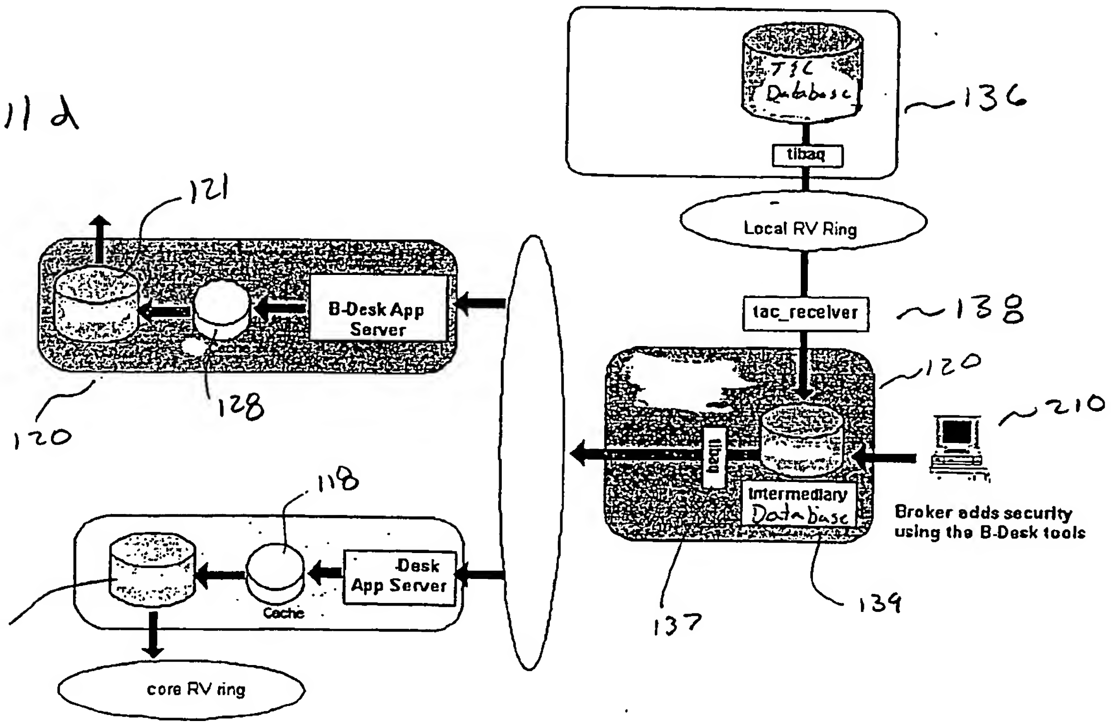


Figure 11d



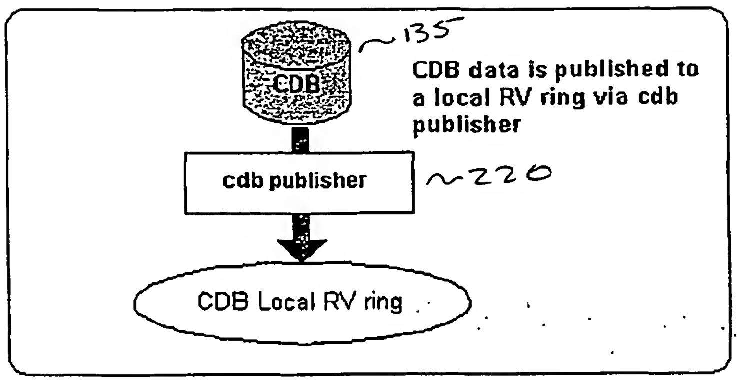


Figure 12a

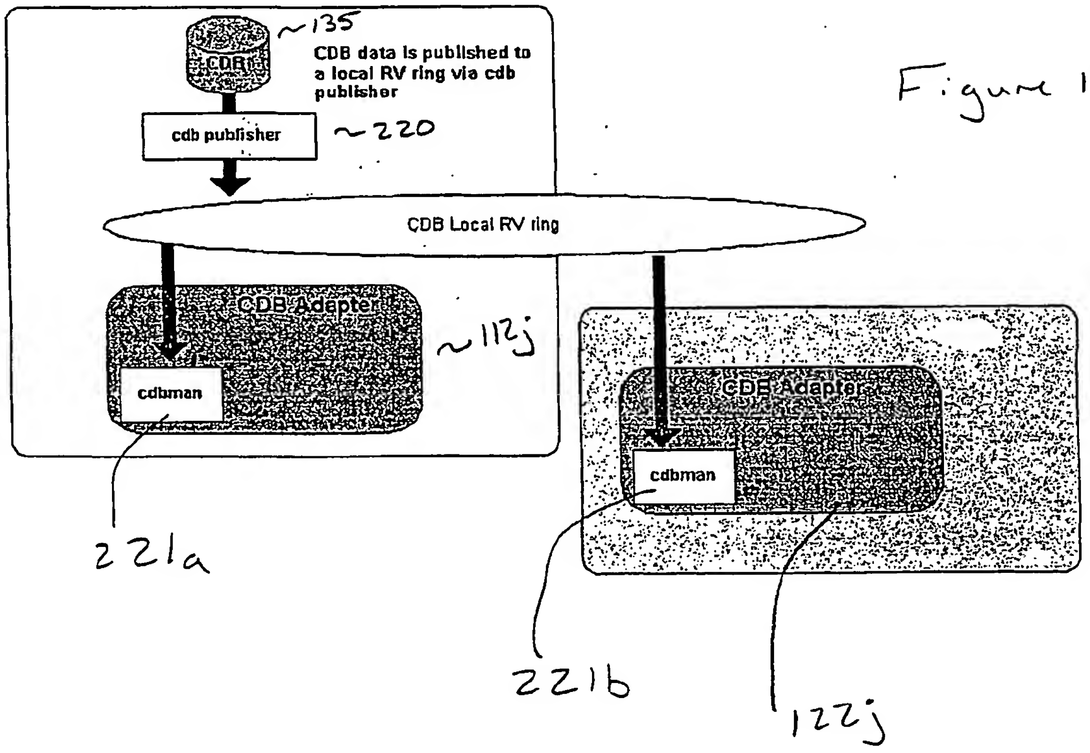


Figure 12b

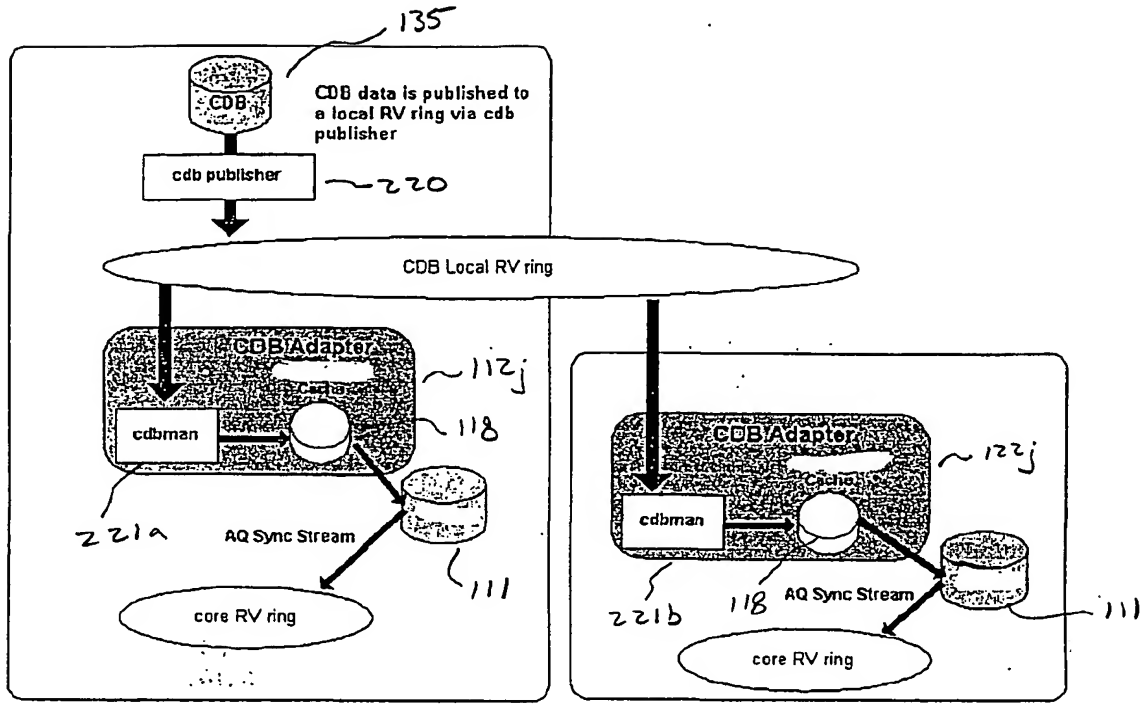

135
CDB
CDB data is published to a local RV ring via cdb publisher
cdb publisher
220
CDB Local RV ring
CDB Adapter
Cache
112j
cdbman
118
221a
AQ Sync Stream
111
core RV ring
CDB Adapter
Cache
122j
cdbman
221b
118
AQ Sync Stream
111
core RV ring

Figure 12c

Figure 13a

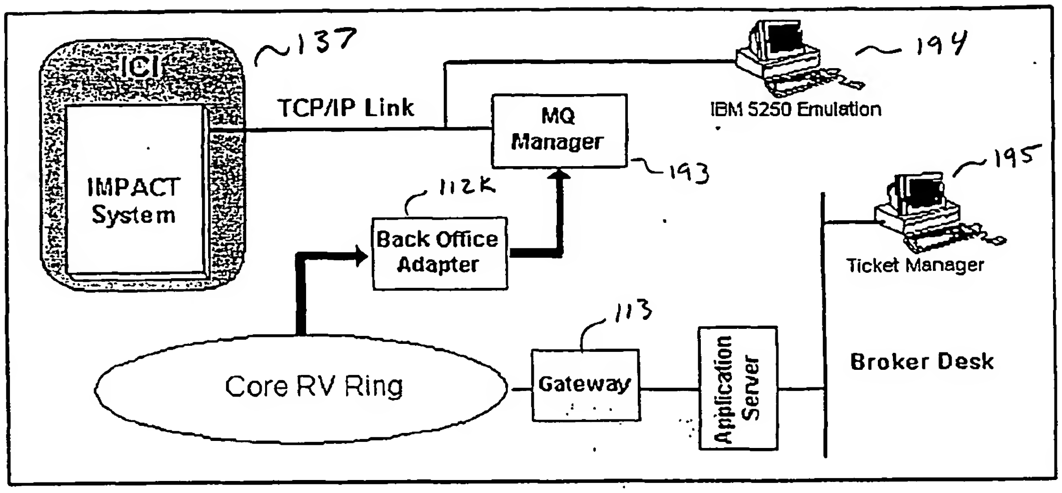
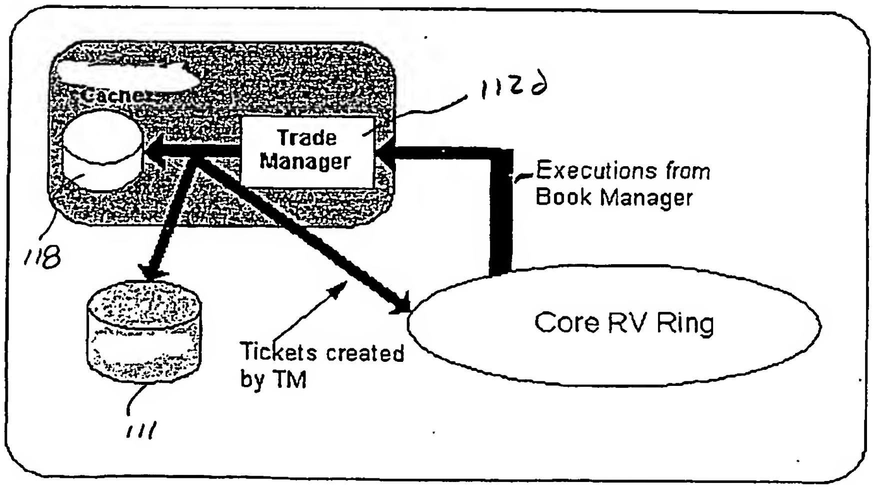


Figure 13b



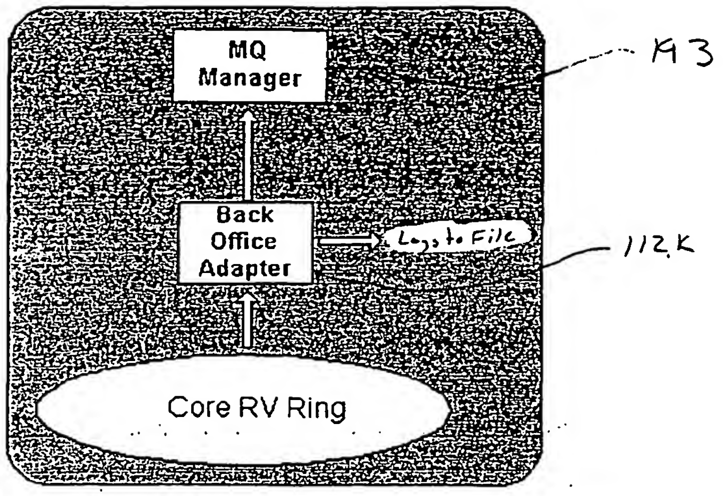
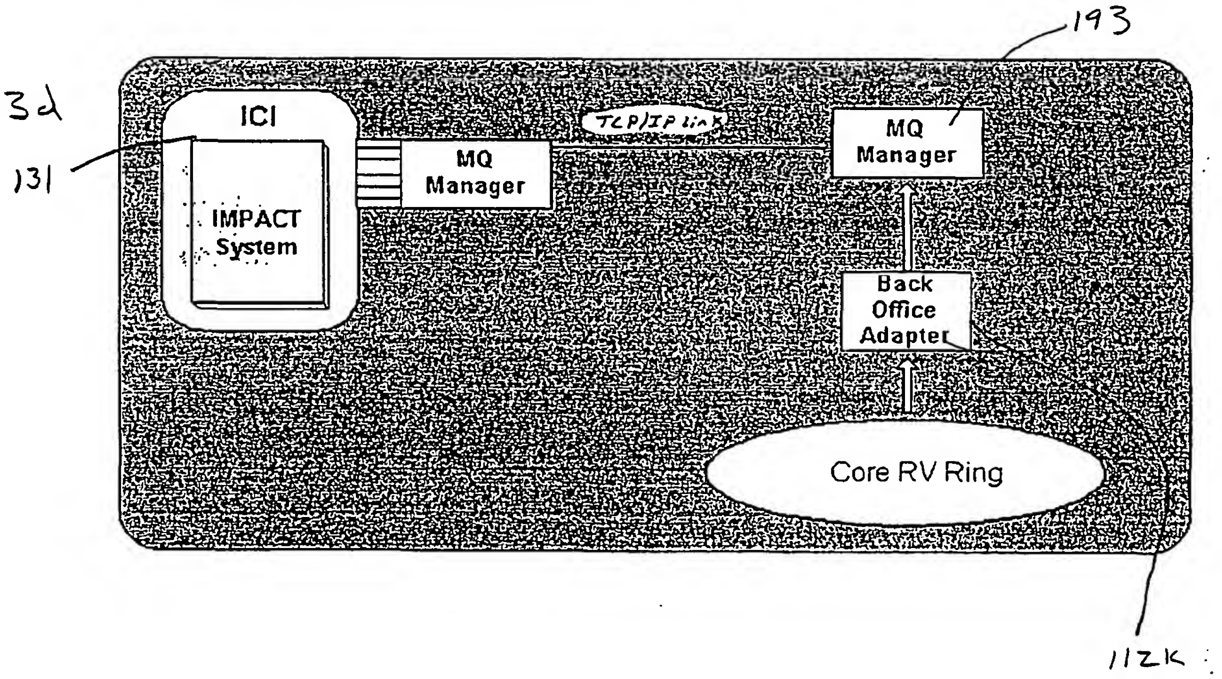


Figure 13c

Figure 13d



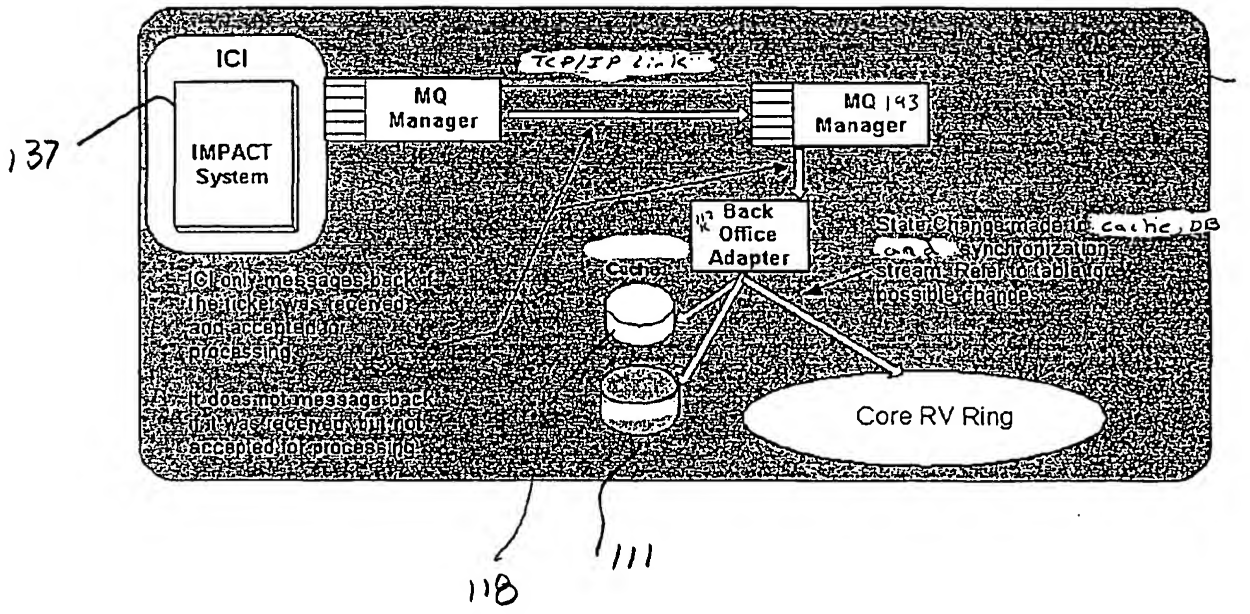

ICI
IMPACT System
137
MQ Manager
TCP/IP Link
MQ 143 Manager
112 Back Office Adapter
Cache
State Change made to cache, DB and synchronization stream. Refer to table for possible changes
ICI only messages back if the ticket was received and accepted for processing
It does not message back if it was received but not accepted for processing
Core RV Ring
118
111

Figure 13e.

Figure 14a

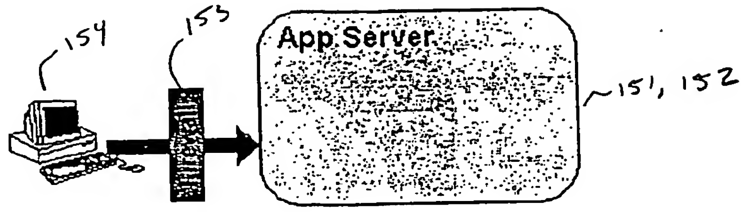
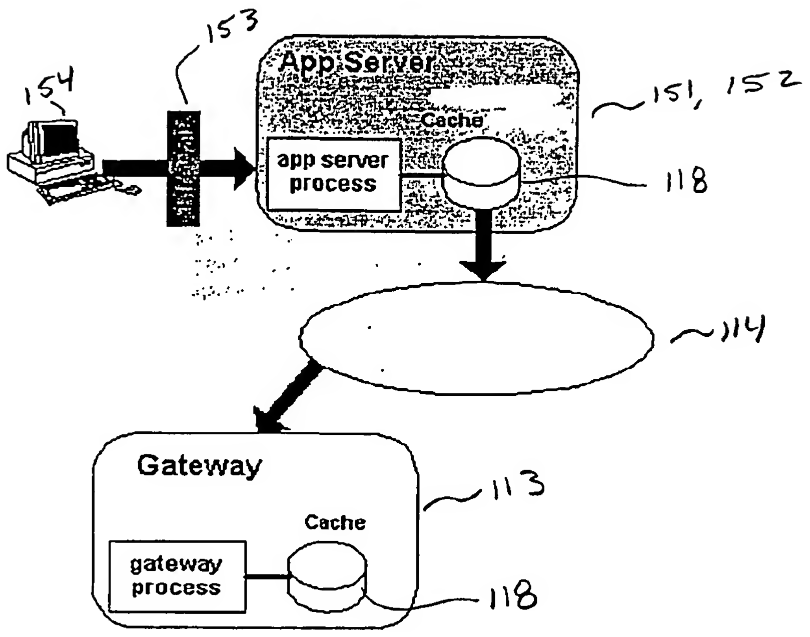


Figure 14b



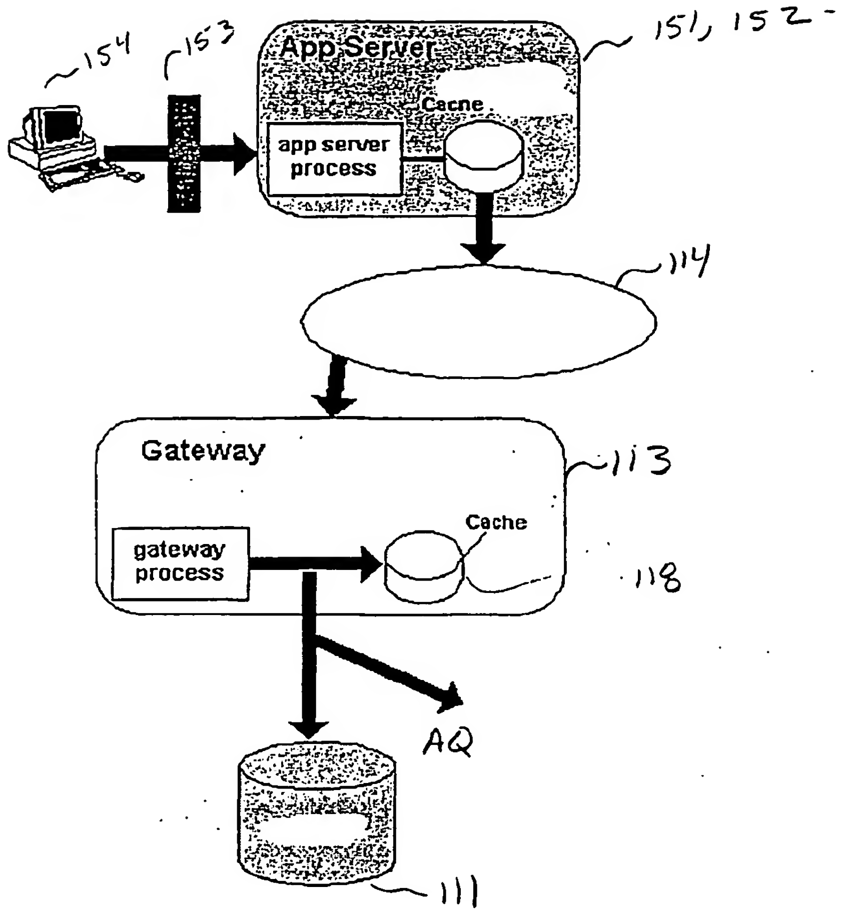
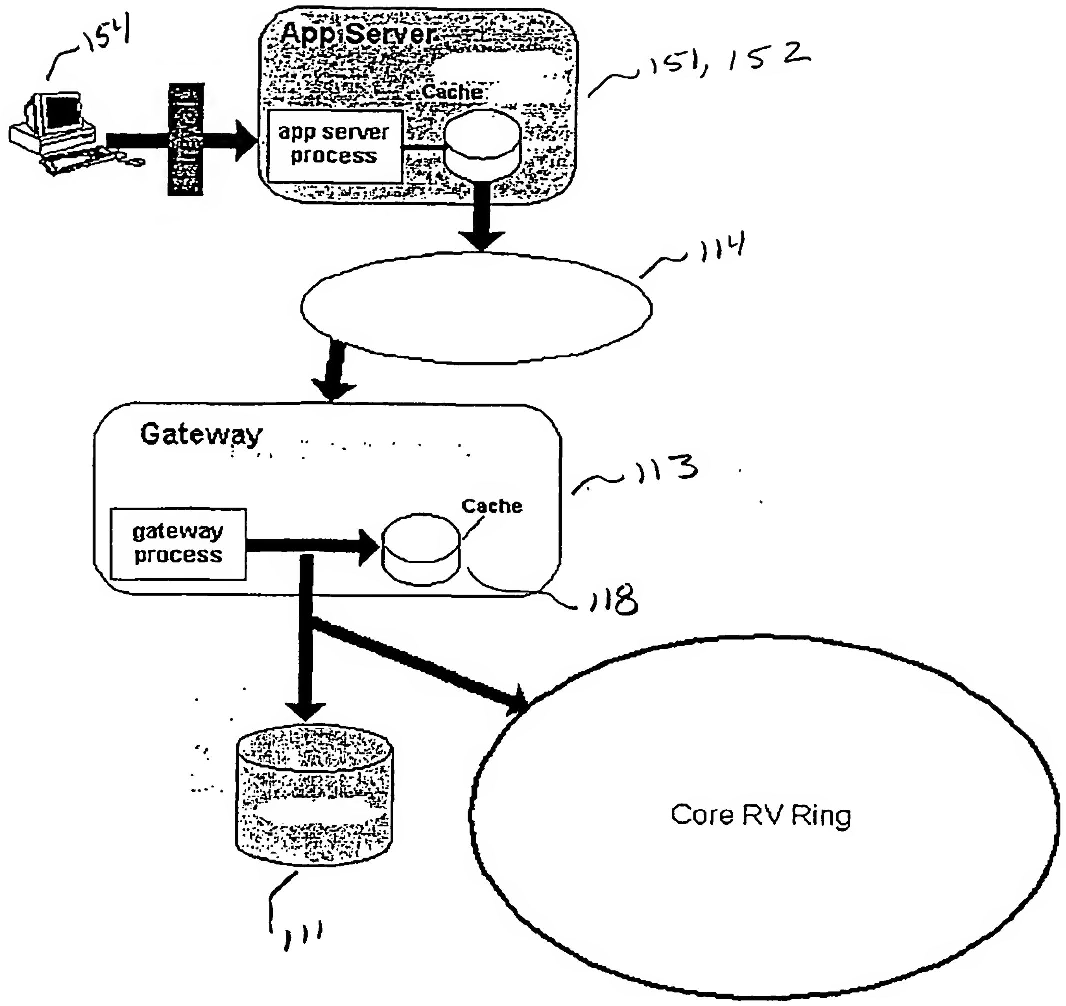

154
153
App Server
151, 152
Cacne
app server process
114
Gateway
113
Cache
gateway process
118
AQ
111

Figure 14c

Figure 14d


154
AppServer
Cache
app server process
151, 152
114
Gateway
113
Cache
gateway process
118
Core RV Ring
111

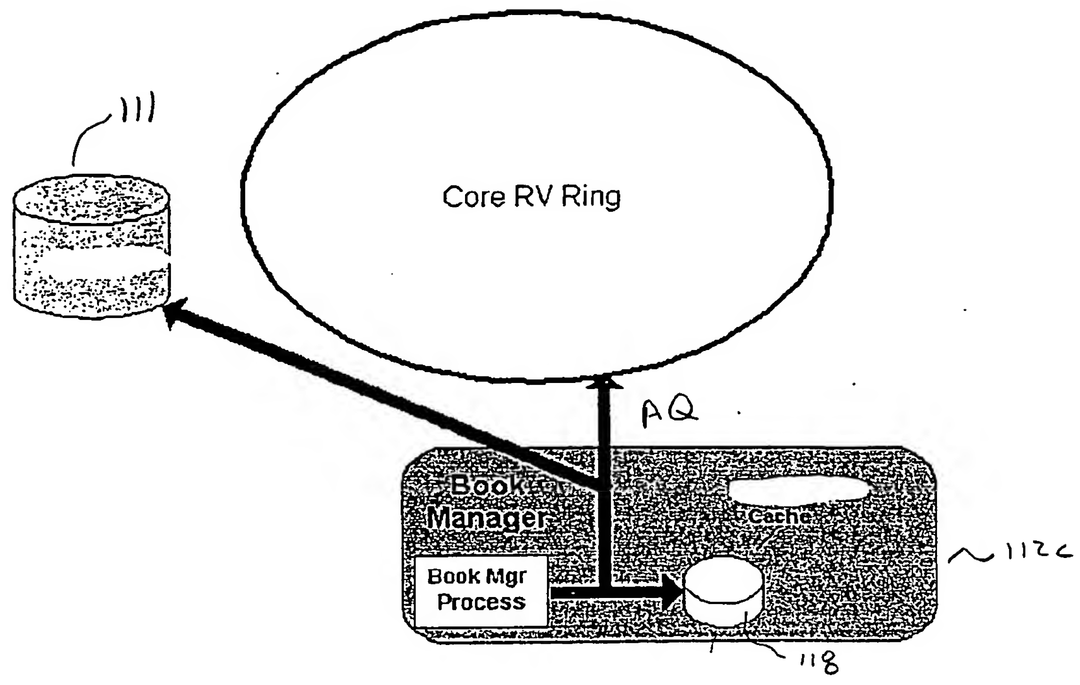

111
Core RV Ring
AQ
Book Manager
Cache
Book Mgr Process
112c
118

Figure 14e

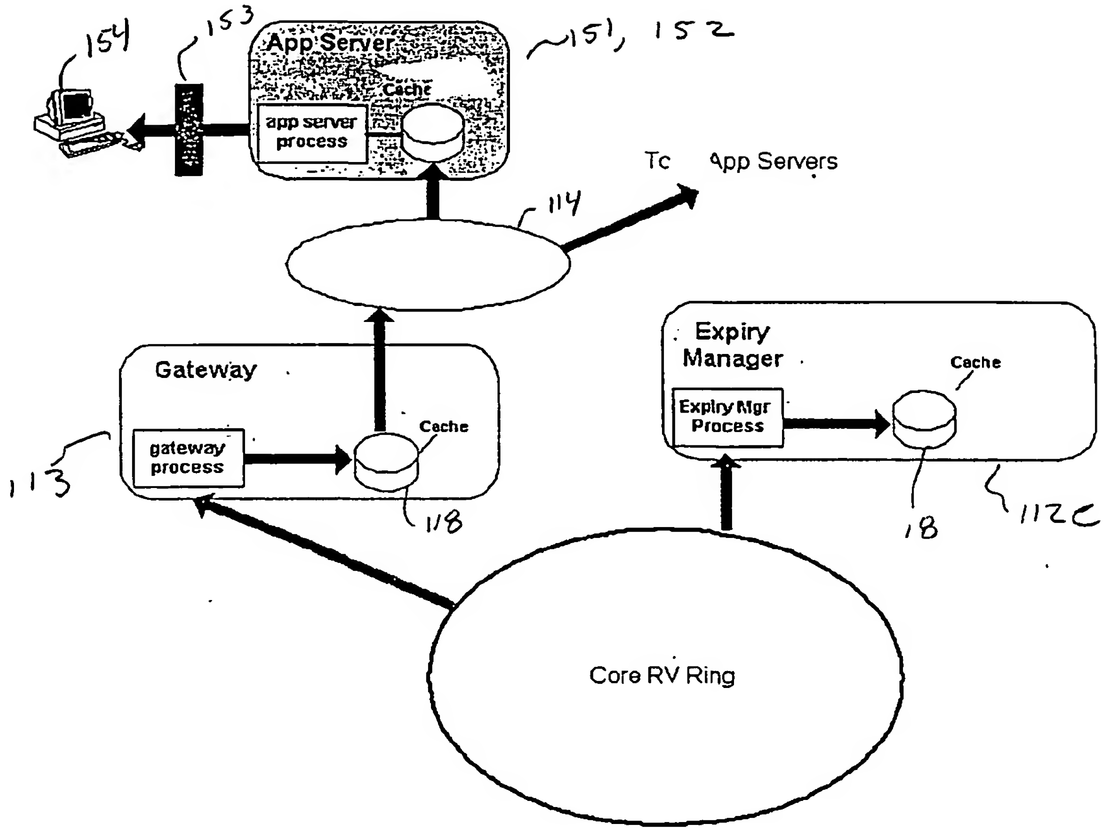

154
153
App Server
Cache
app server process
151, 152
To App Servers
114
Gateway
gateway process
Cache
113
118
Expiry Manager
Expiry Mgr Process
Cache
18
112c
Core RV Ring

Figure 14F

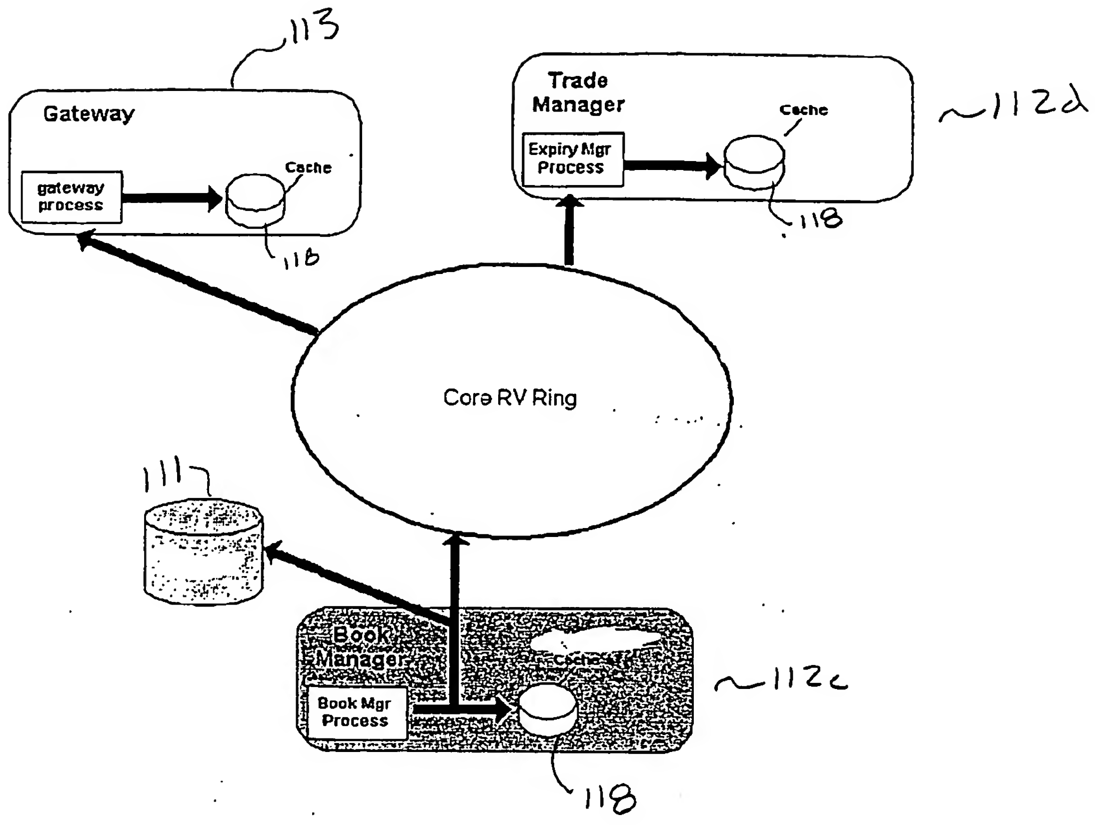

113
Gateway
gateway process
Cache
116
Trade Manager
Expiry Mgr Process
Cache
118
112d
Core RV Ring
111
Book Manager
Book Mgr Process
Cache
118
112c

Figure 14g

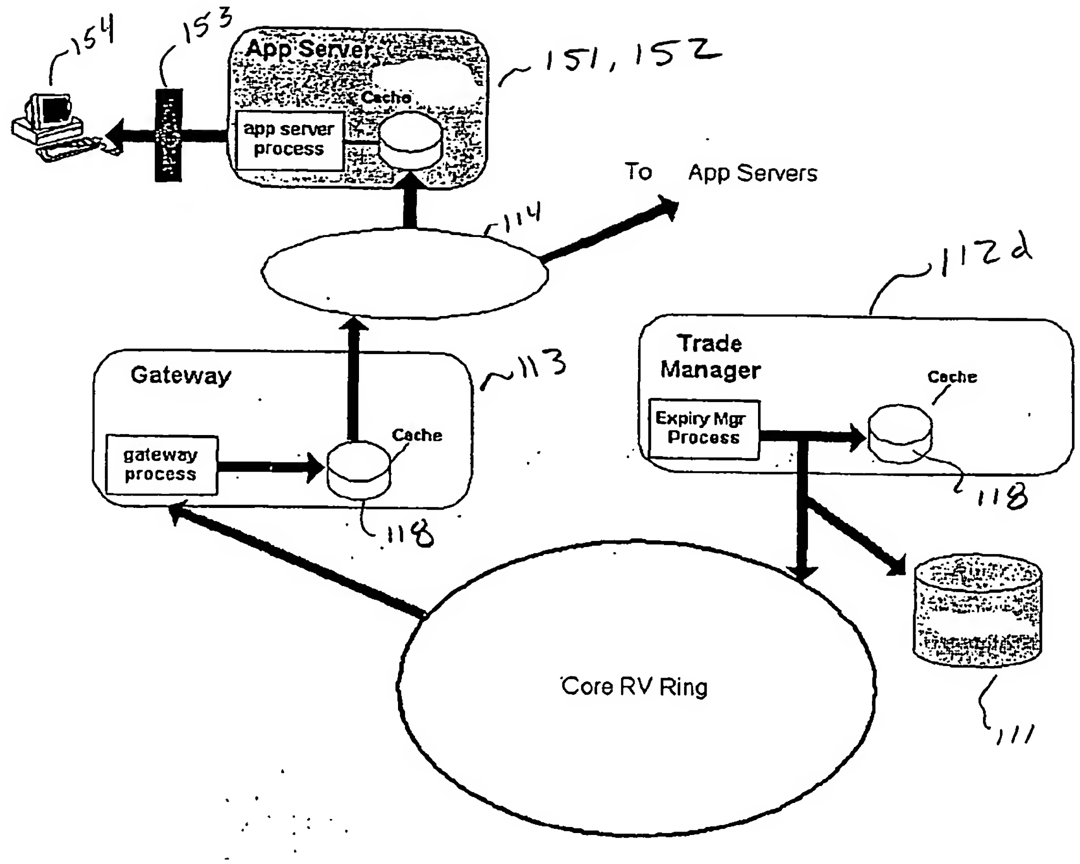

154
153
App Server
Cache
app server process
151, 152
To App Servers
114
112d
Trade Manager
Cache
Expiry Mgr Process
118
Gateway
113
gateway process
Cache
118
Core RV Ring
111

Figure 14h

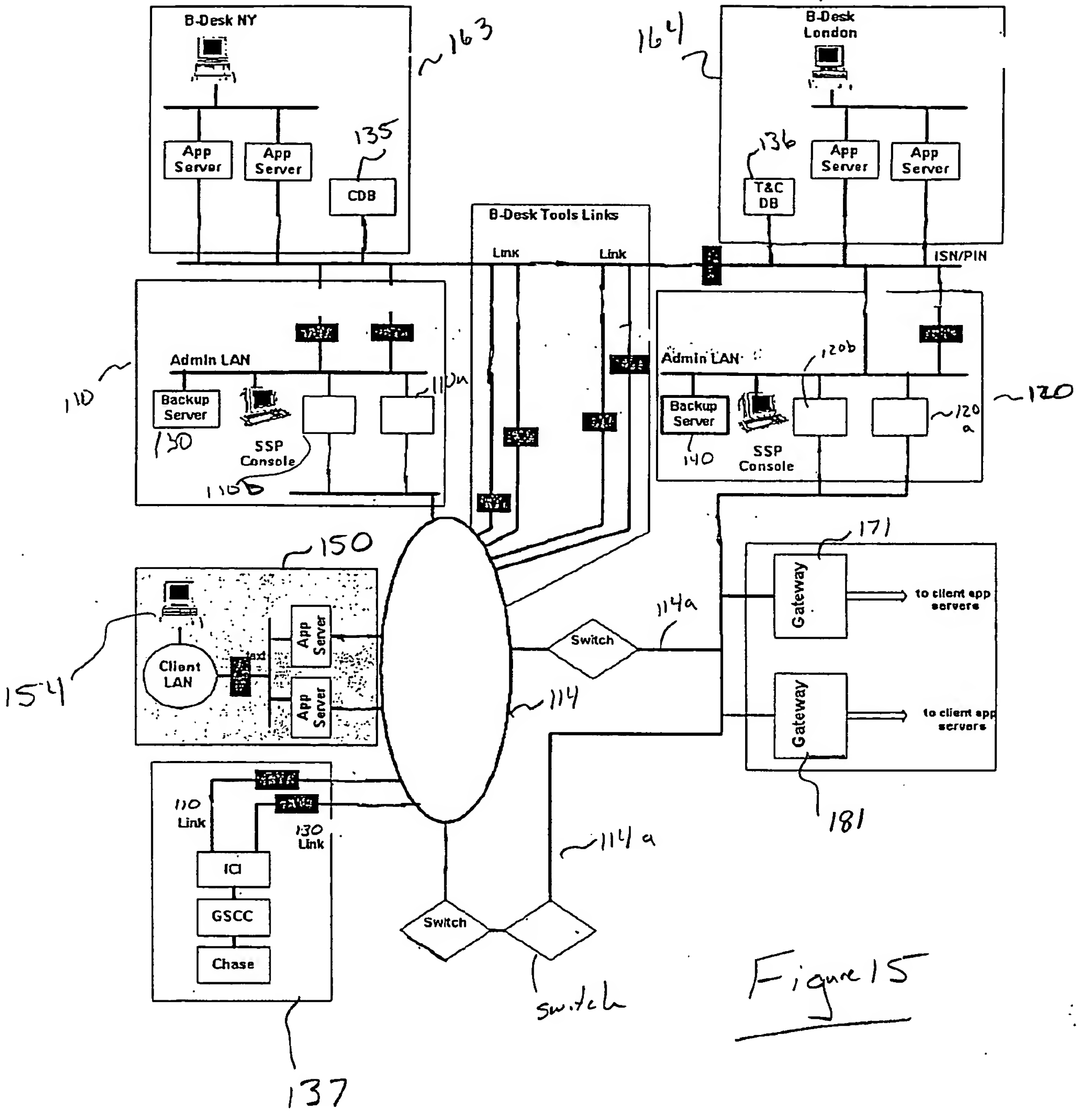
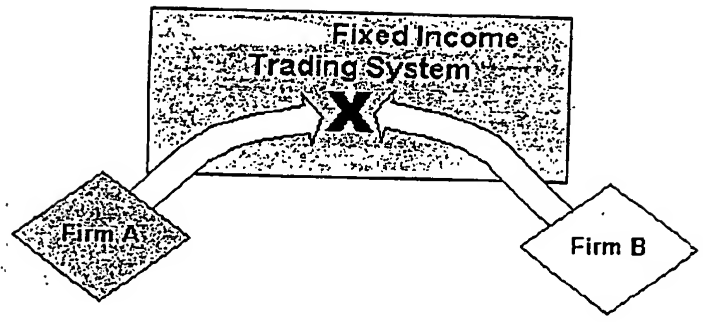

B-Desk NY
App Server
App Server
CDB
135
163
B-Desk London
164
136
T&C DB
App Server
App Server
B-Desk Tools Links
Link
Link
ISN/PIN
Admin LAN
110
Backup Server
130
SSP Console
110a
110b
Admin LAN
120b
Backup Server
140
SSP Console
120a
120
150
Client LAN
App Server
App Server
154
114
114a
Switch
171
Gateway
to client app servers
Gateway
to client app servers
181
110 Link
130 Link
ICI
GSCC
Chase
137
Switch
switch
114a

Figure 15

Figure 16



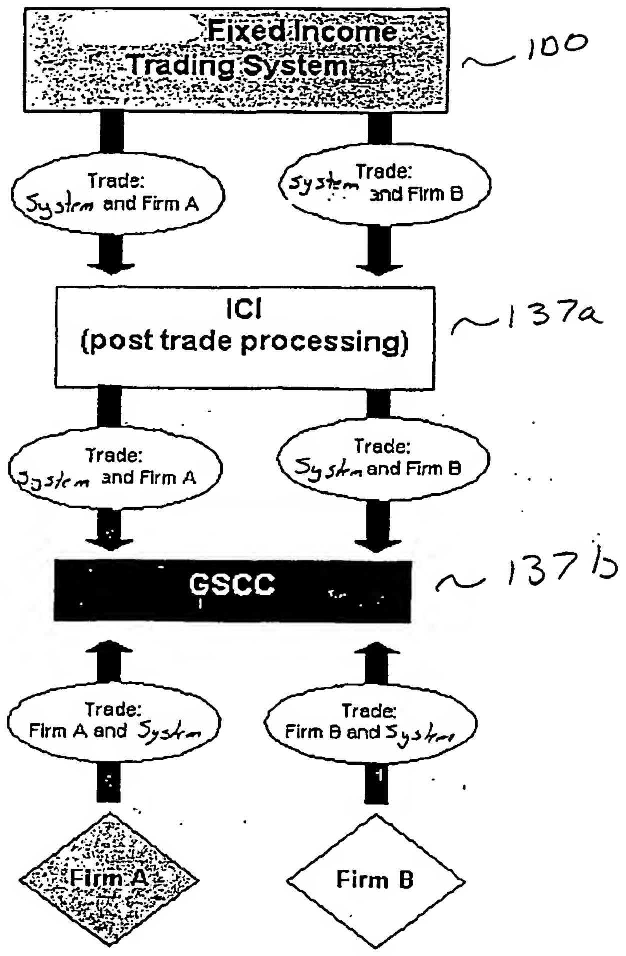

Fixed Income Trading System
100
Trade: System and Firm A
Trade: System and Firm B
ICI (post trade processing)
137a
Trade: System and Firm A
Trade: System and Firm B
GSCC
137b
Trade: Firm A and System
Trade: Firm B and System
Firm A
Firm B

Figure 17a

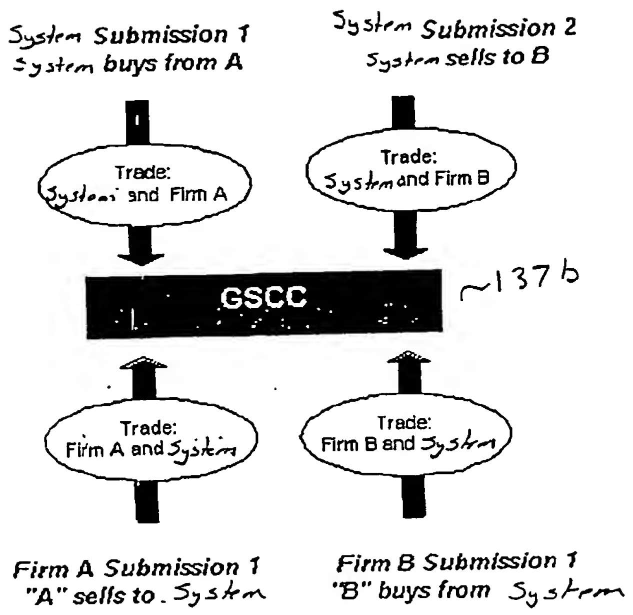

System Submission 1
System buys from A
System Submission 2
System sells to B
Trade:
System and Firm A
Trade:
System and Firm B
GSCC
~137b
Trade:
Firm A and System
Trade:
Firm B and System
Firm A Submission 1
"A" sells to System
Firm B Submission 1
"B" buys from System

Figure 17b

Figure 18a

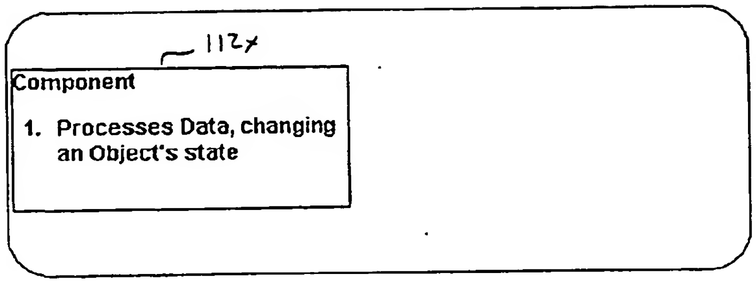


Figure 18b

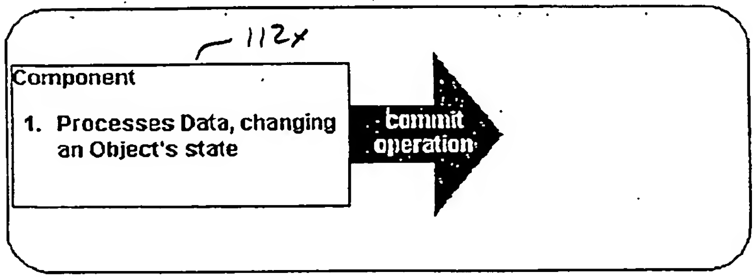


Figure 18c

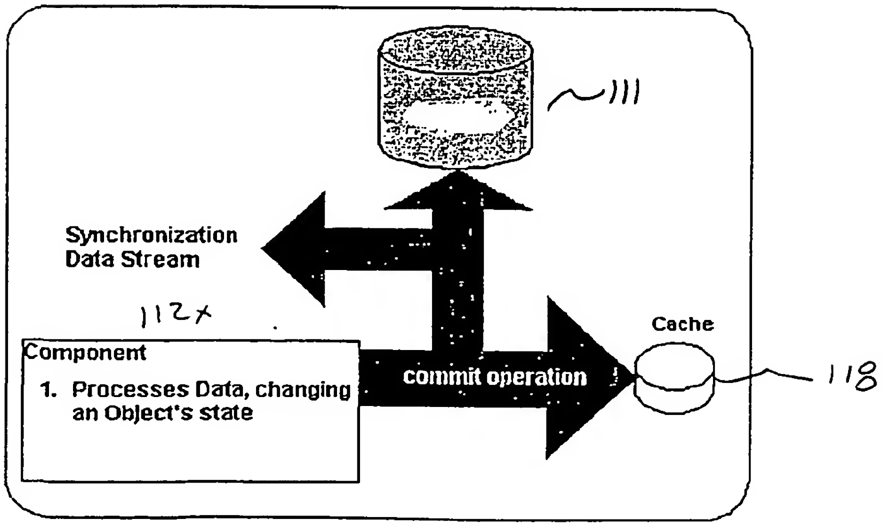

111
Synchronization Data Stream
112x
Component
1. Processes Data, changing an Object's state
commit operation
Cache
118

Figure 19

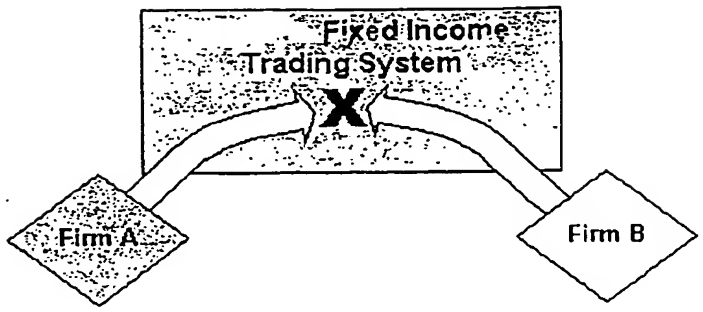
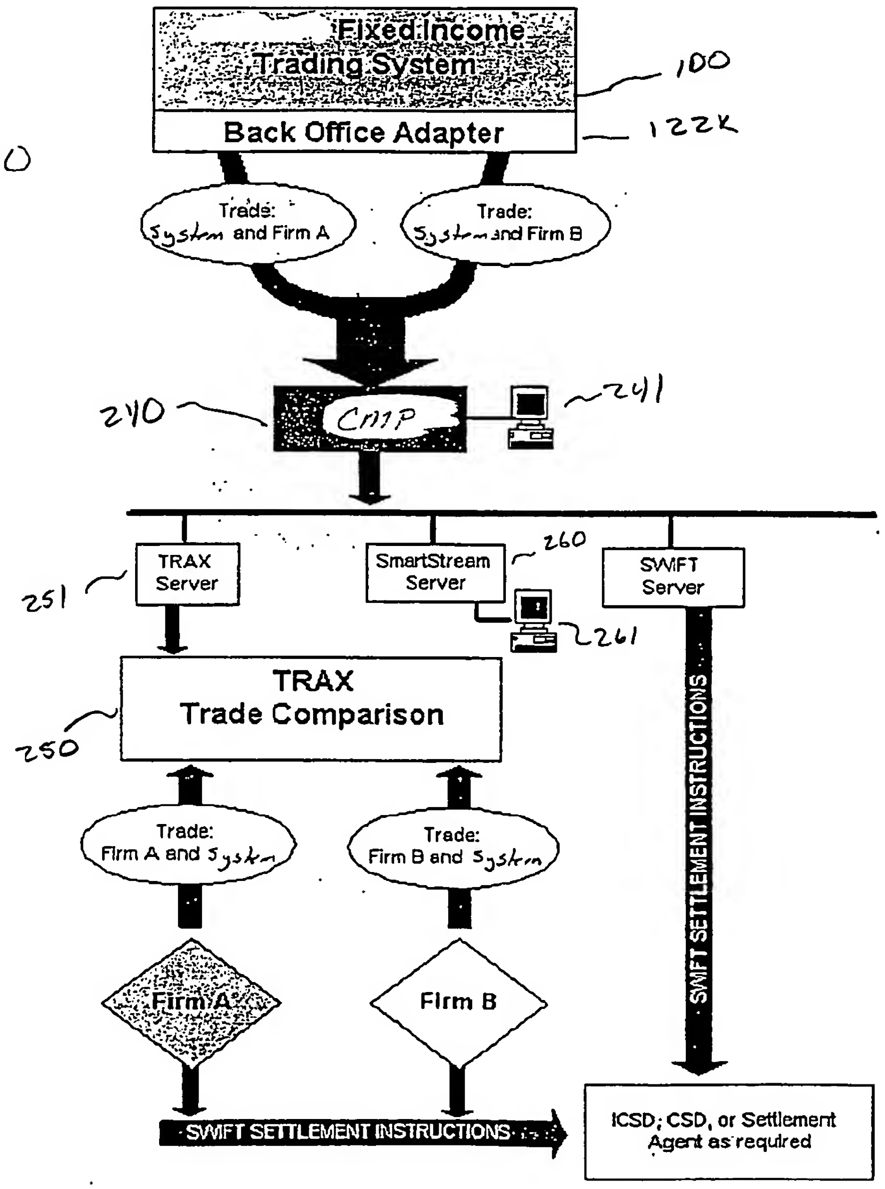


Figure 20



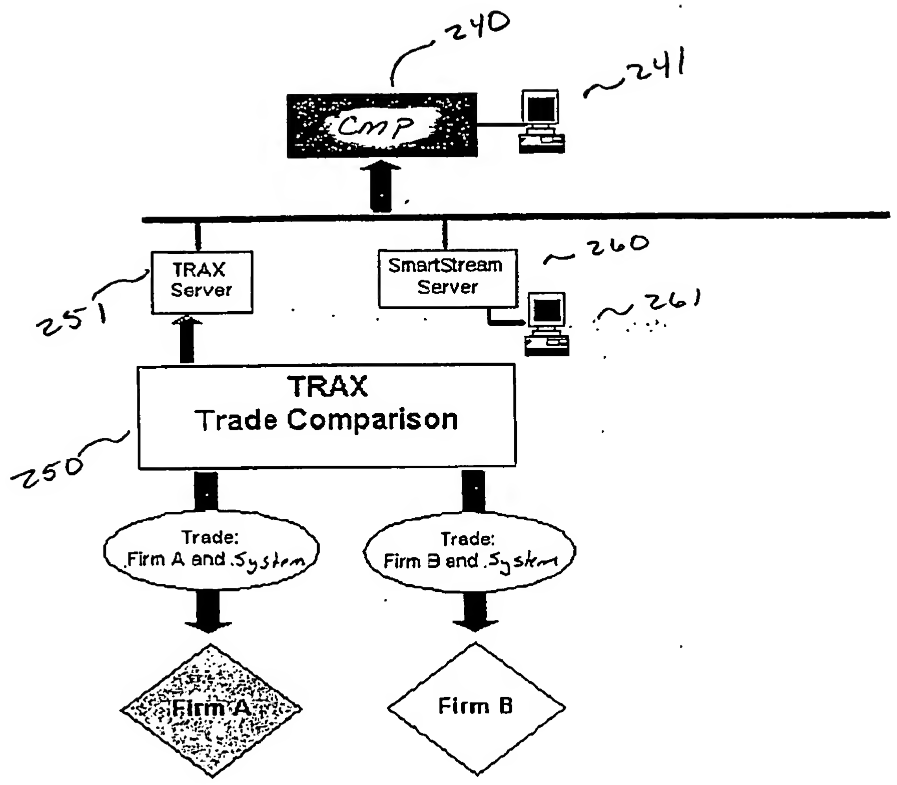

240
CMP
241
TRAX Server
251
SmartStream Server
260
261
TRAX Trade Comparison
250
Trade: Firm A and System
Trade: Firm B and System
Firm A
Firm B

Figure 21

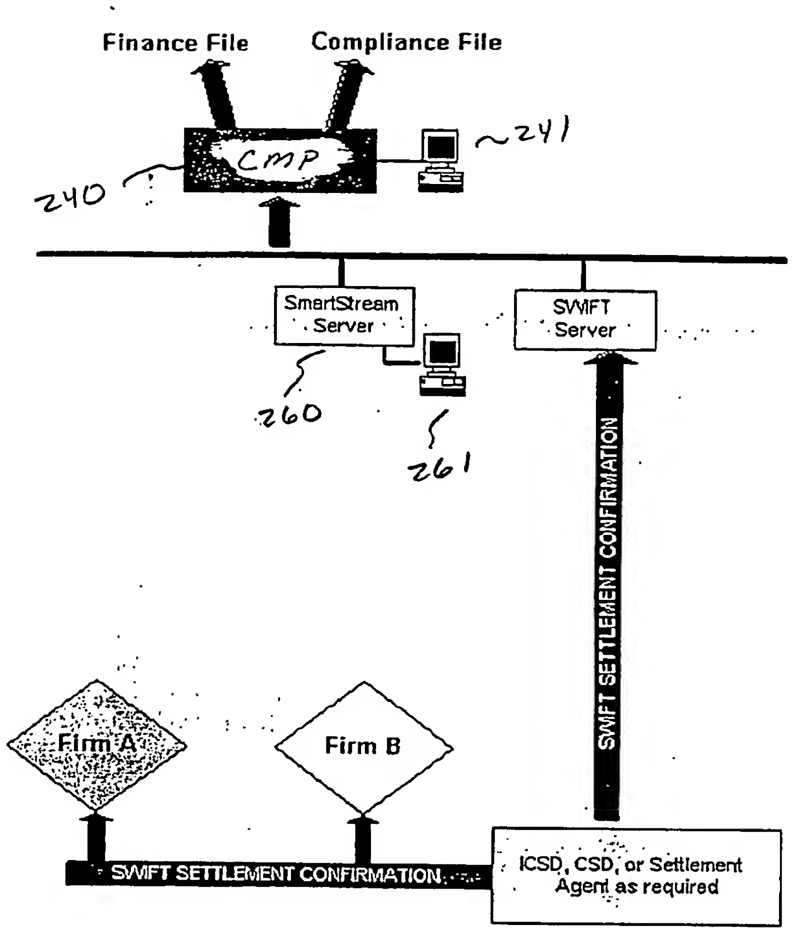

Finance File
Compliance File
CMP
241
240
SmartStream Server
SWIFT Server
260
261
SWIFT SETTLEMENT CONFIRMATION
Firm A
Firm B
SWIFT SETTLEMENT CONFIRMATION
ICSD, CSD, or Settlement Agent as required

Figure 22

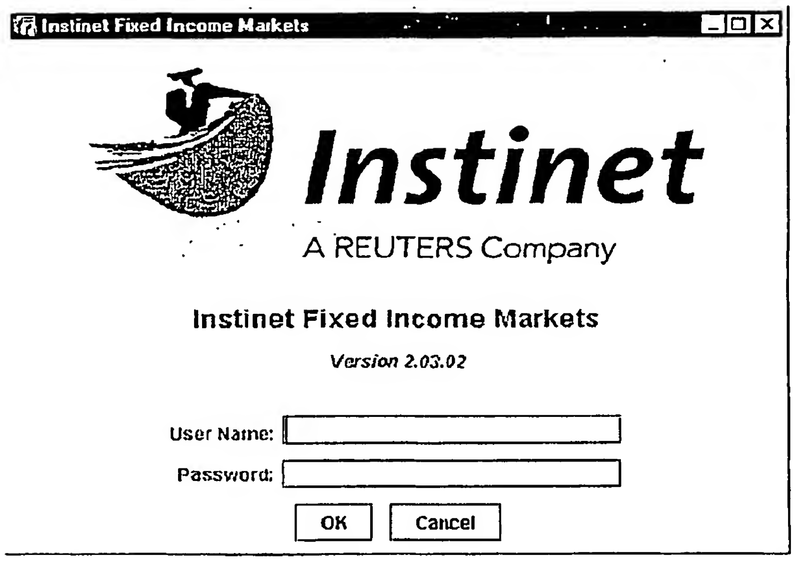
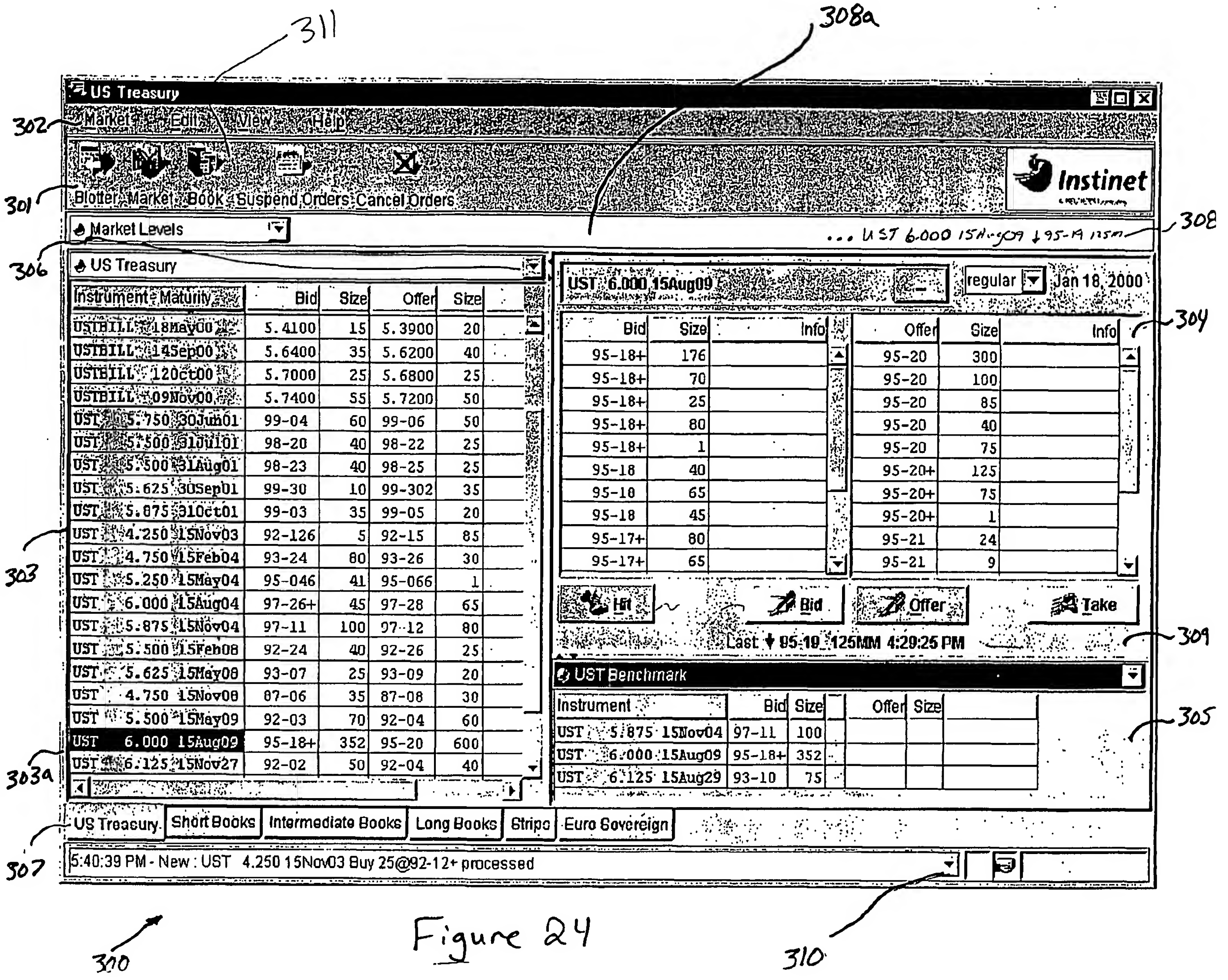

Instinet Fixed Income Markets
Instinet
A REUTERS Company
Instinet Fixed Income Markets
Version 2.03.02
User Name:
Password:
OK
Cancel

Figure 23

4



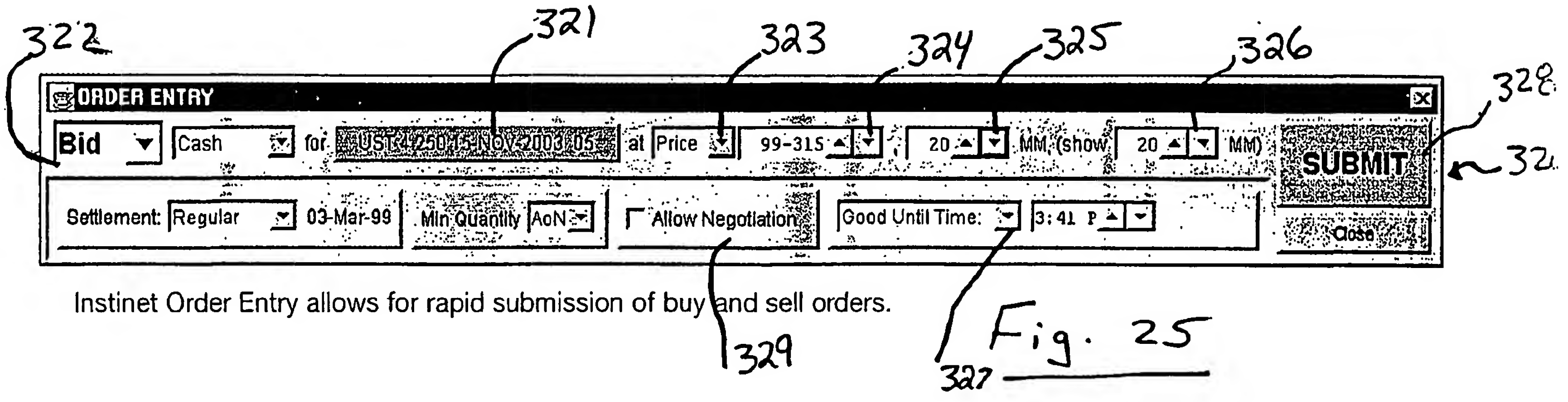
Instrument Maturity	Bid	Size	Offer	Size
USTBILL 18May00	5.4100	15	5.3900	20
USTBILL 14Sep00	5.6400	35	5.6200	40
USTBILL 12Oct00	5.7000	25	5.6800	25
USTBILL 09Nov00	5.7400	55	5.7200	50
UST 5.750 30Jun01	99-04	60	99-06	50
UST 5.500 31Jul01	98-20	40	98-22	25
UST 5.500 31Aug01	98-23	40	98-25	25
UST 5.625 30Sep01	99-30	10	99-302	35
UST 5.875 31Oct01	99-03	35	99-05	20
UST 4.250 15Nov03	92-126	5	92-15	85
UST 4.750 15Feb04	93-24	80	93-26	30
UST 5.250 15May04	95-046	41	95-066	1
UST 6.000 15Aug04	97-26+	45	97-28	65
UST 5.875 15Nov04	97-11	100	97-12	80
UST 5.500 15Feb08	92-24	40	92-26	25
UST 5.625 15May08	93-07	25	93-09	20
UST 4.750 15Nov08	87-06	35	87-08	30
UST 5.500 15May09	92-03	70	92-04	60
UST 6.000 15Aug09	95-18+	352	95-20	600
UST 6.125 15Nov27	92-02	50	92-04	40

Bid	Size	Info	Offer	Size	Info
95-18+	176		95-20	300	
95-18+	70		95-20	100	
95-18+	25		95-20	85	
95-18+	80		95-20	40	
95-18+	1		95-20	75	
95-18	40		95-20+	125	
95-18	65		95-20+	75	
95-18	45		95-20+	1	
95-17+	80		95-21	24	
95-17+	65		95-21	9	

Instrument	Bid	Size	Offer	Size
UST 5.875 15Nov04	97-11	100		
UST 6.000 15Aug09	95-18+	352		
UST 6.125 15Aug29	93-10	75		

5:40:39 PM - New : UST 4.250 15Nov03 Buy 25@92-12+ processed

Figure 24



ORDER ENTRY

Bid | Cash for UST 4.250 15-Nov-2003 05 at Price 99-315 20 MM (show 20 MM)

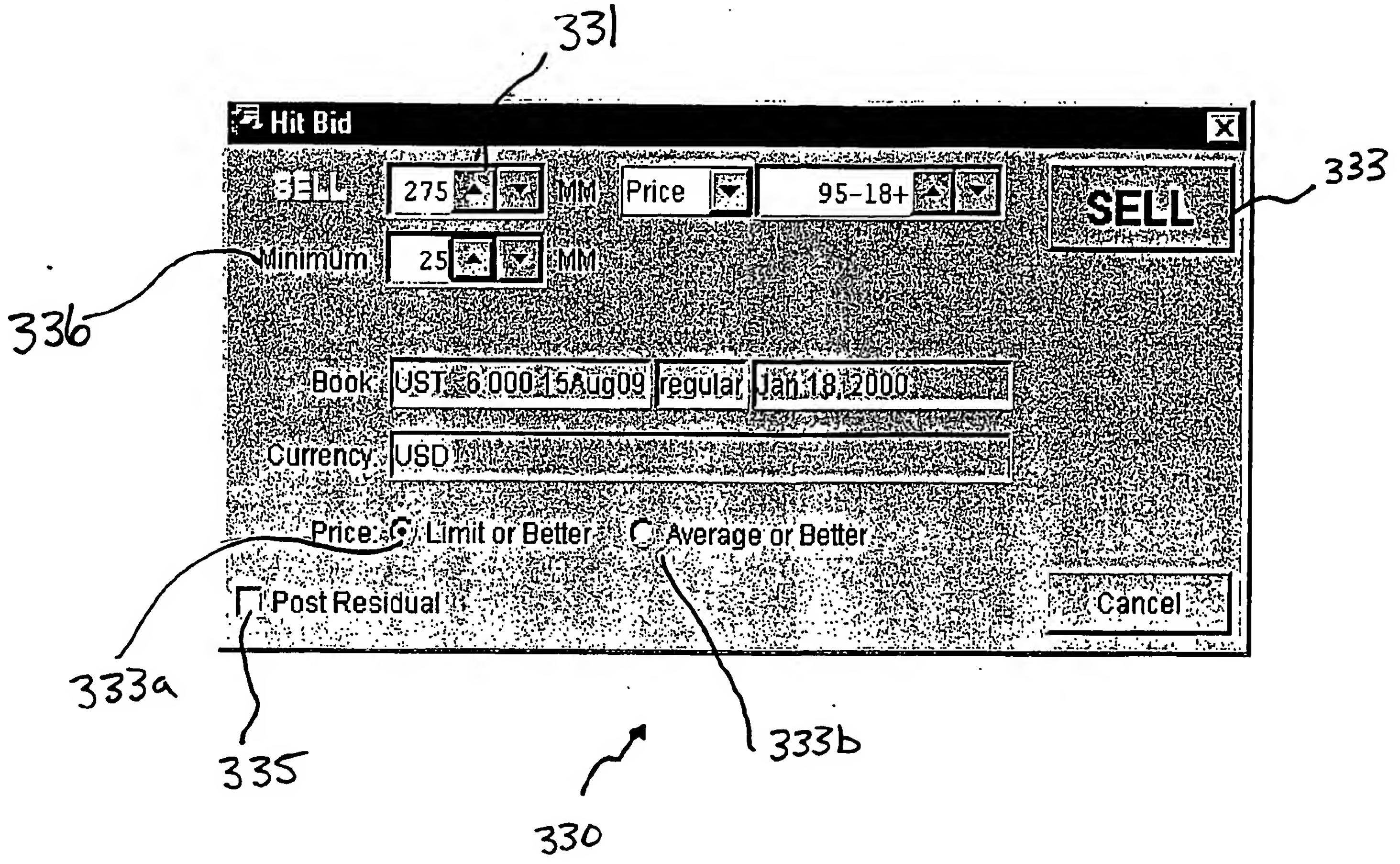
SUBMIT

Settlement: Regular 03-Mar-99 Min Quantity AoN ☐ Allow Negotiation Good Until Time: 3:41 P

Close

Instinet Order Entry allows for rapid submission of buy and sell orders.

Fig. 25



SELL 275 MM Price 95-18+

Minimum 25 MM

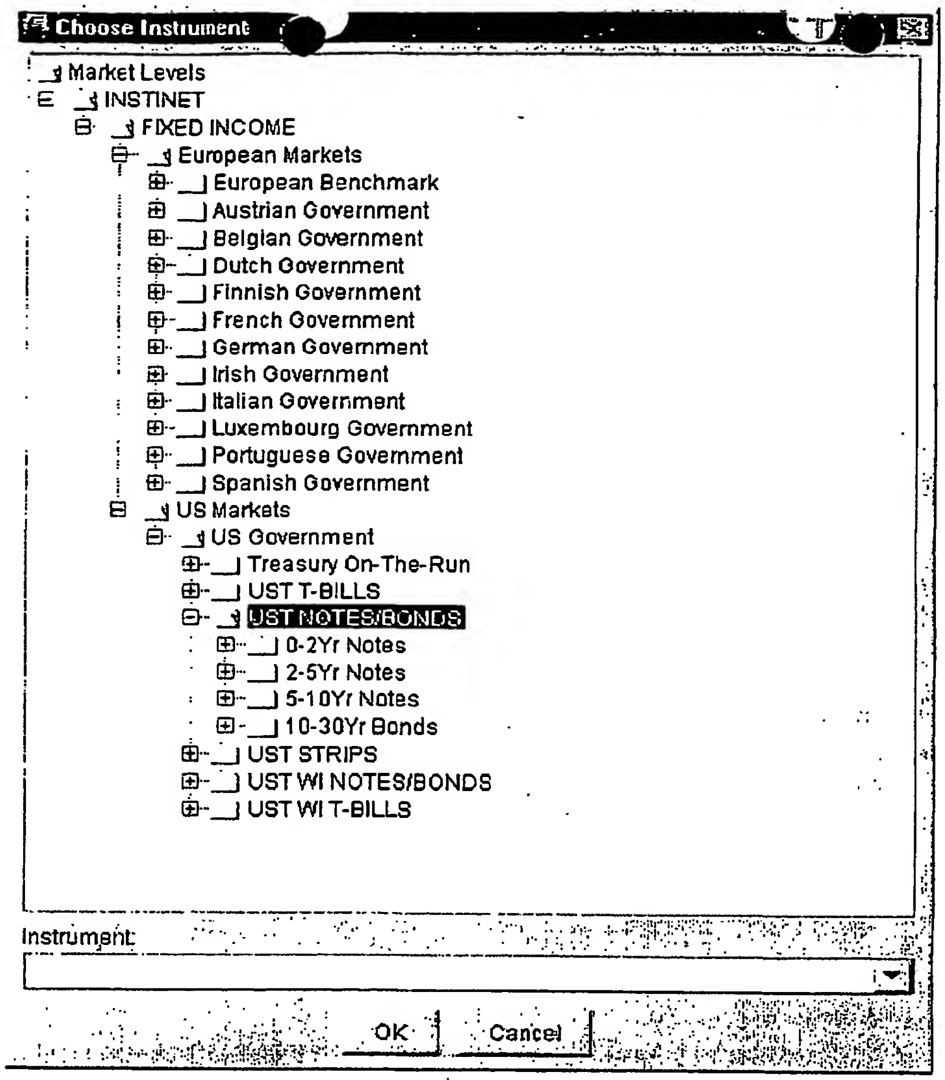
Book: UST 6.000 15Aug09 regular Jan 18 2000

Currency: USD

Price: [✓ Limit or Better] Average or Better

Post Residual

Figure 26



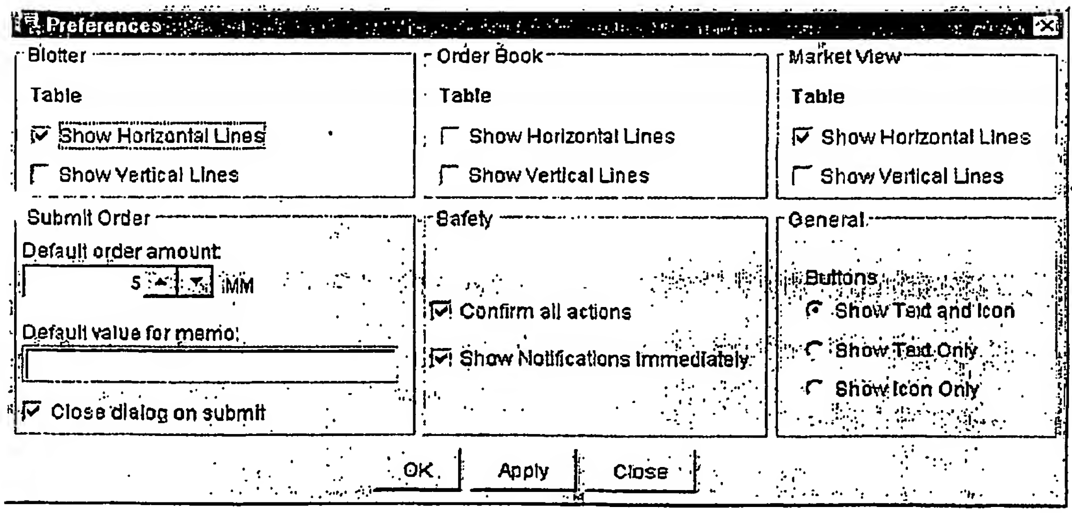
Choose Instrument

- Market Levels
- INSTINET
  - FIXED INCOME
    - European Markets
      - European Benchmark
      - Austrian Government
      - Belgian Government
      - Dutch Government
      - Finnish Government
      - French Government
      - German Government
      - Irish Government
      - Italian Government
      - Luxembourg Government
      - Portuguese Government
      - Spanish Government
    - US Markets
      - US Government
        - Treasury On-The-Run
        - UST T-BILLS
        - UST NOTES/BONDS
          - 0-2Yr Notes
          - 2-5Yr Notes
          - 5-10Yr Notes
          - 10-30Yr Bonds
        - UST STRIPS
        - UST WI NOTES/BONDS
        - UST WI T-BILLS

Instrument:

306

Figure 27



Preferences

Blotter
Table
[✓] Show Horizontal Lines
[ ] Show Vertical Lines

Order Book
Table
[ ] Show Horizontal Lines
[ ] Show Vertical Lines

Market View
Table
[✓] Show Horizontal Lines
[ ] Show Vertical Lines

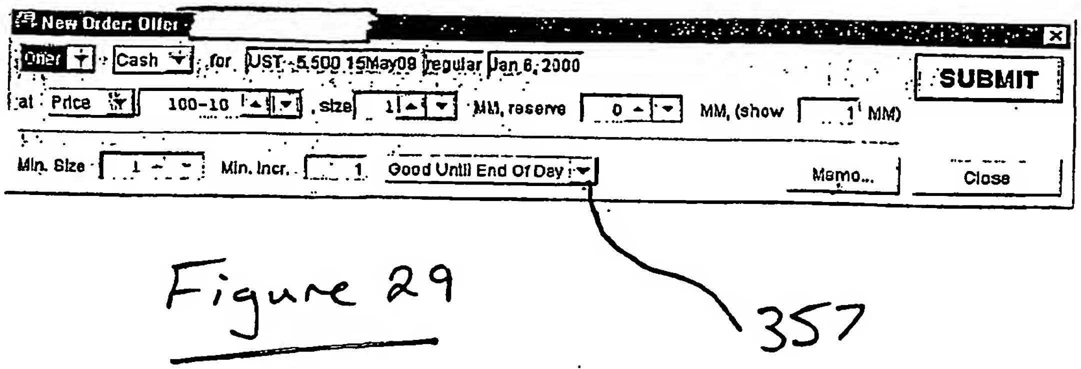
Submit Order
Default order amount: 5 MM
Default value for memo:
[✓] Close dialog on submit

Safety
[✓] Confirm all actions
[✓] Show Notifications Immediately

General
Buttons
[✓ Show Text and Icon]
Show Text Only
Show Icon Only

315

Figure 28

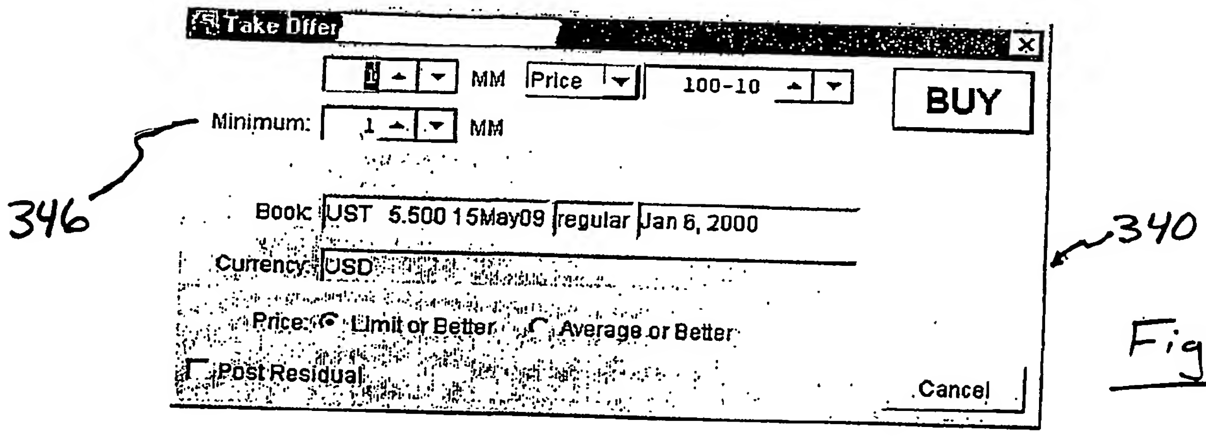

New Order: Offer

Offer | Cash | for UST 5.500 15May09 | regular | Jan 6, 2000 | SUBMIT

at Price | 100-10 | , size 1 | MM, reserve 0 | MM, (show 1 MM)

Min. Size 1 | Min. Incr. 1 | Good Until End Of Day | Memo... | Close

Figure 29


Take Offer

1 MM | Price | 100-10 | BUY

Minimum: 1 MM

Book: UST 5.500 15May09 | regular | Jan 6, 2000

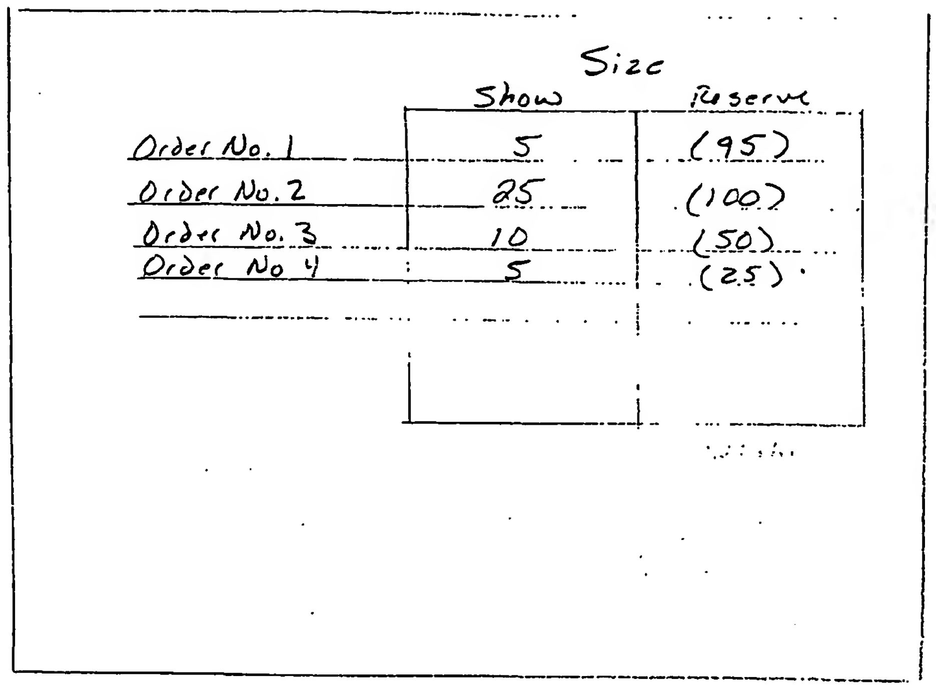
Currency: USD

Price: [✓ Limit or Better] Average or Better

Post Residual

Cancel

Figure 30



	Size	
	Show	Reserve
Order No. 1	5	(95)
Order No. 2	25	(100)
Order No. 3	10	(50)
Order No 4	5	(25)

Figure 31

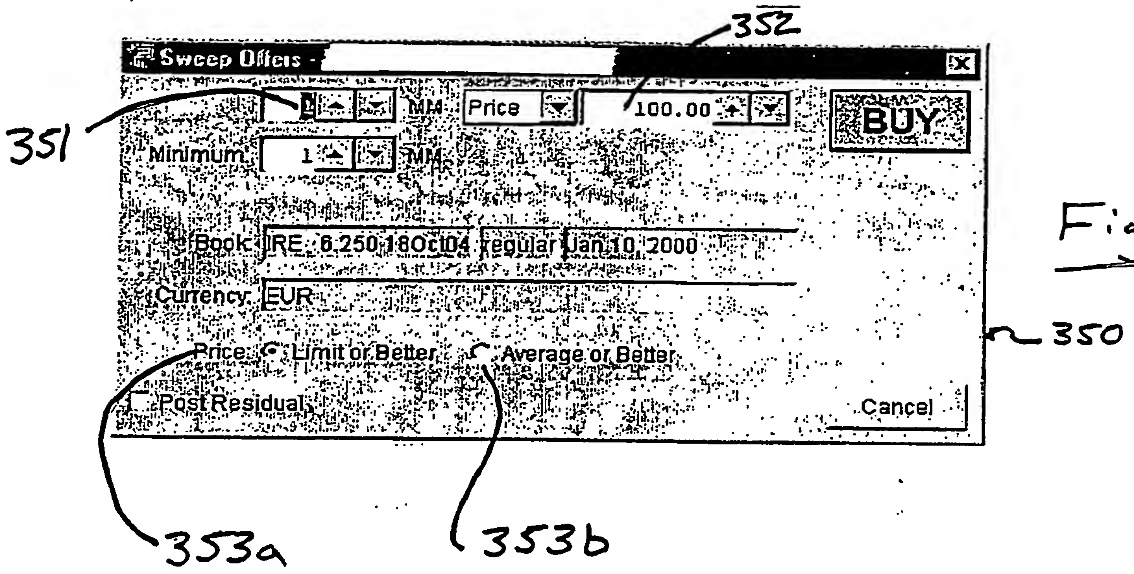


Figure 32

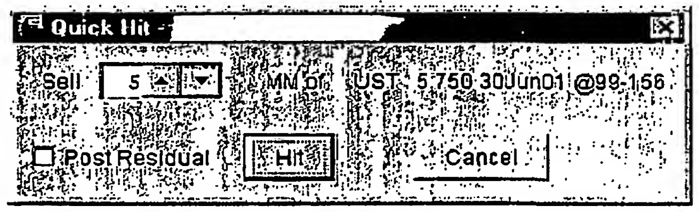


Figure 33

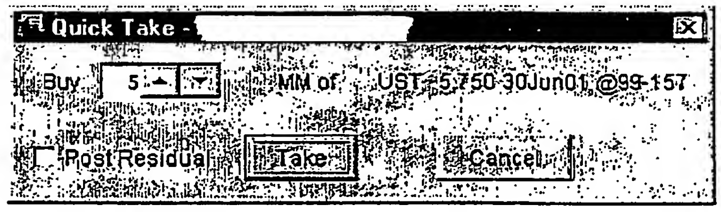


Figure 34

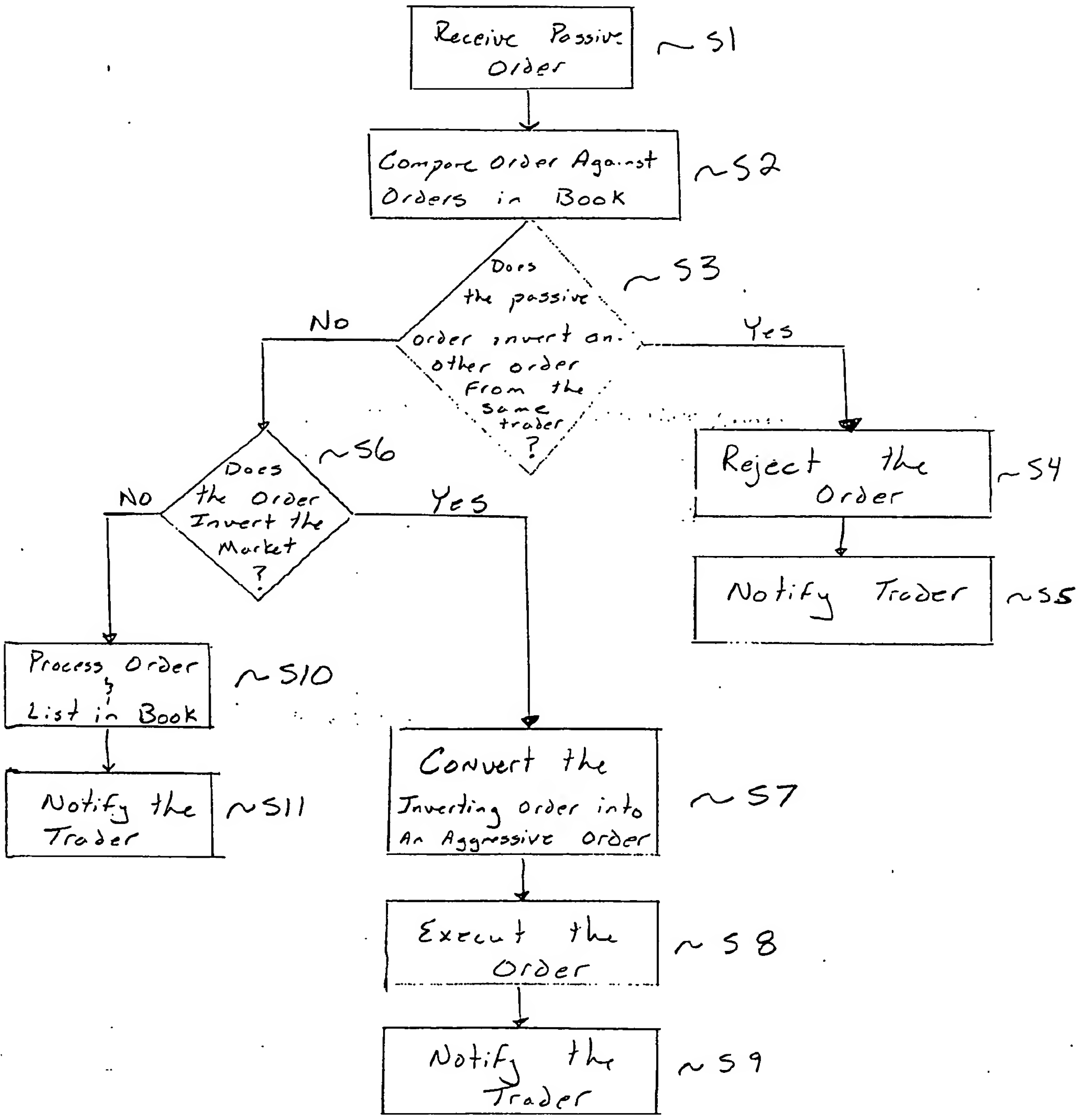
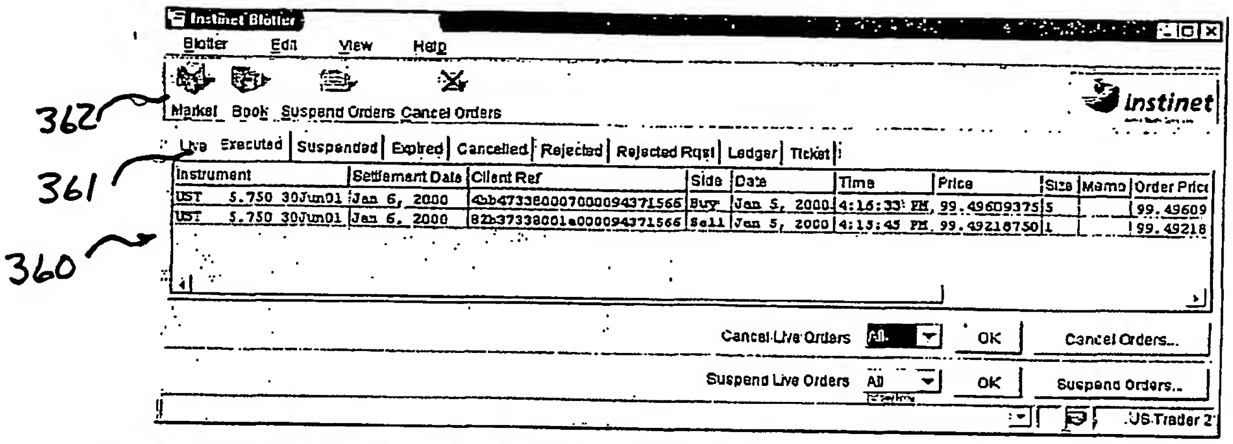

Receive Passive Order
S1
Compare Order Against Orders in Book
S2
Does the passive order invert another order from the same trader?
S3
No
Yes
Reject the Order
S4
Notify Trader
S5
Does the Order Invert the Market?
S6
No
Yes
Process Order & List in Book
S10
Notify the Trader
S11
Convert the Inverting order into An Aggressive Order
S7
Execute the Order
S8
Notify the Trader
S9

Figure 35


Instinet Blotter
Blotter Edit View Help
Market Book Suspend Orders Cancel Orders
Instinet
Live Executed Suspended Expired Cancelled Rejected Rejected Rqst Ledger Ticket
Cancel Live Orders All OK Cancel Orders...
Suspend Live Orders All OK Suspend Orders...
US Trader 2
362
361
360

Instrument	Settlement Date	Client Ref	Side	Date	Time	Price	Size	Memo	Order Price
UST 5.750 30Jun01	Jan 6, 2000	4bb473380007000094371566	Buy	Jan 5, 2000	4:16:33 PM	99.49609375	5		99.49609
UST 5.750 30Jun01	Jan 6, 2000	82b37338001a000094371566	Sell	Jan 5, 2000	4:13:45 PM	99.49218750	1		99.49218

Figure 36a

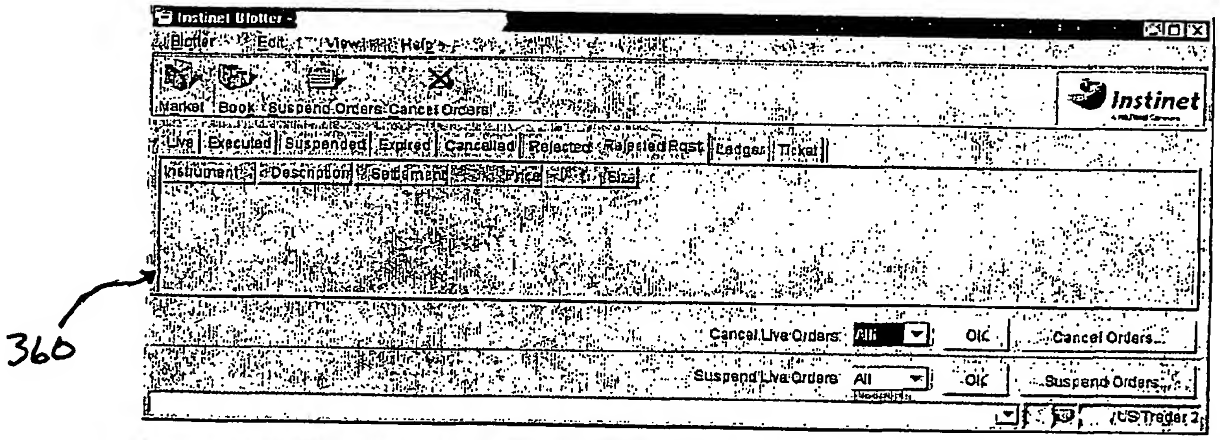
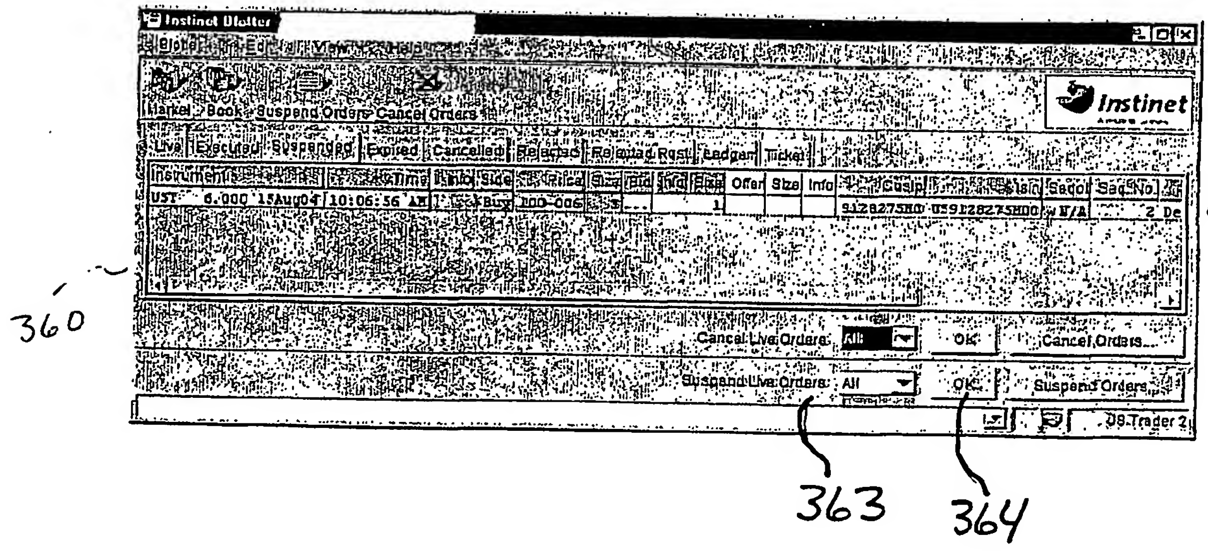
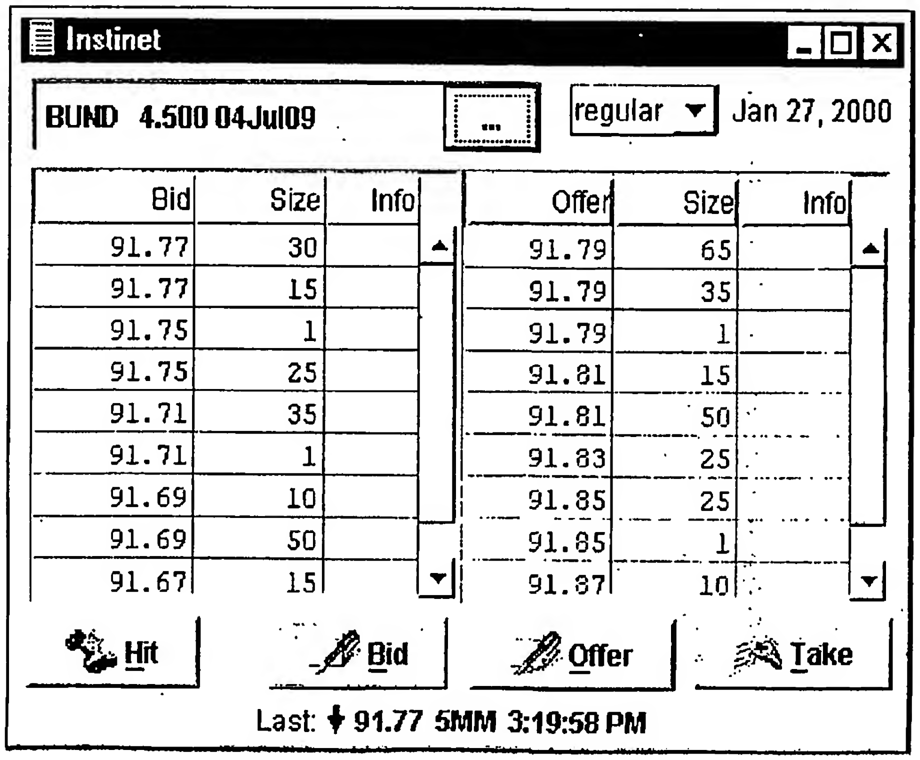

Instinet Blotter
Blotter Edit View Help
Market Book Suspend Orders Cancel Orders
Instinet
Live Executed Suspended Expired Cancelled Rejected Rejected Rqst Ledger Ticket
Instrument Description Settlement Price Size
Cancel Live Orders All OK Cancel Orders...
Suspend Live Orders All OK Suspend Orders...
US Trader 2
360

Figure 36b


Instinet Blotter
Blotter Edit View Help
Market Book Suspend Orders Cancel Orders
Instinet
Live Executed Suspended Expired Cancelled Rejected Rejected Rqst Ledger Ticket
Cancel Live Orders All OK Cancel Orders...
Suspend Live Orders All OK Suspend Orders...
US Trader 2
360
363
364

Instrument	Time	Info	Side	Price	Size	Bid	Info	Size	Offer	Size	Info	Cusip	Isin	Sedol	Seq No.	[illegible]
UST 6.000 15Aug04	10:06:56 AM	1	Buy	100-006	[illegible]	---		1				9128275H0	US9128275H00	N/A	2	De

Figure 36c



BUND 4.500 04Jul09

Jan 27, 2000

Bid	Size	Info	Offer	Size	Info
91.77	30		91.79	65	
91.77	15		91.79	35	
91.75	1		91.79	1	
91.75	25		91.81	15	
91.71	35		91.81	50	
91.71	1		91.83	25	
91.69	10		91.85	25	
91.69	50		91.85	1	
91.67	15		91.87	10	

Last: ↓ 91.77 5MM 3:19:58 PM

370

Figure 37

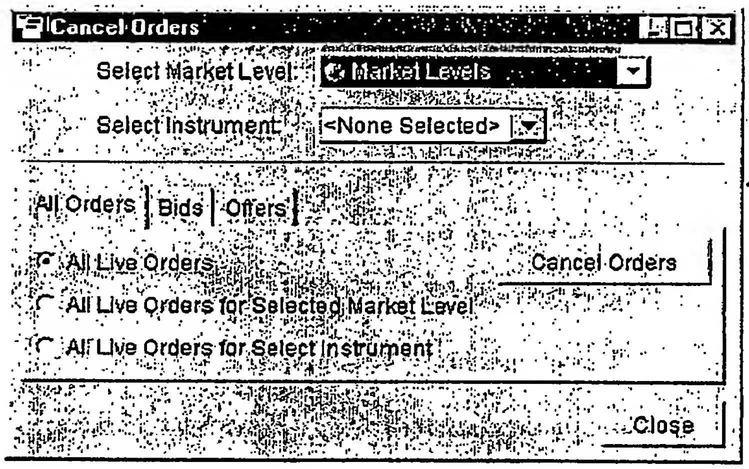


Figure 38a

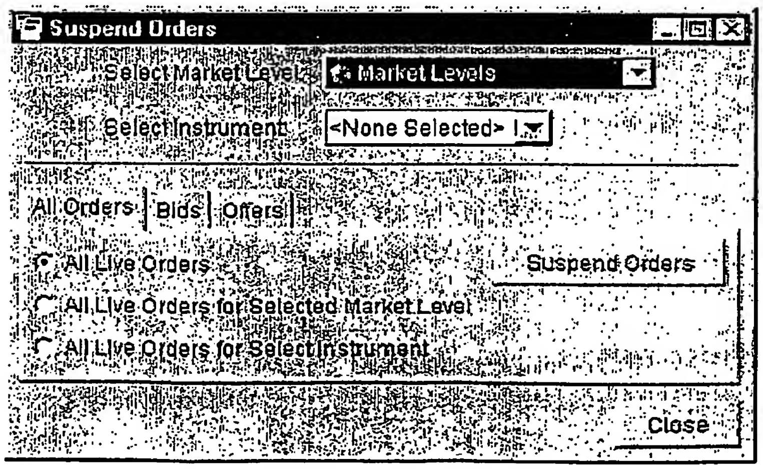
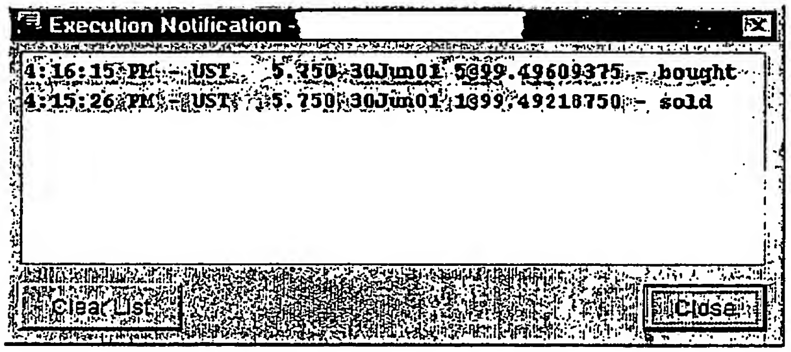


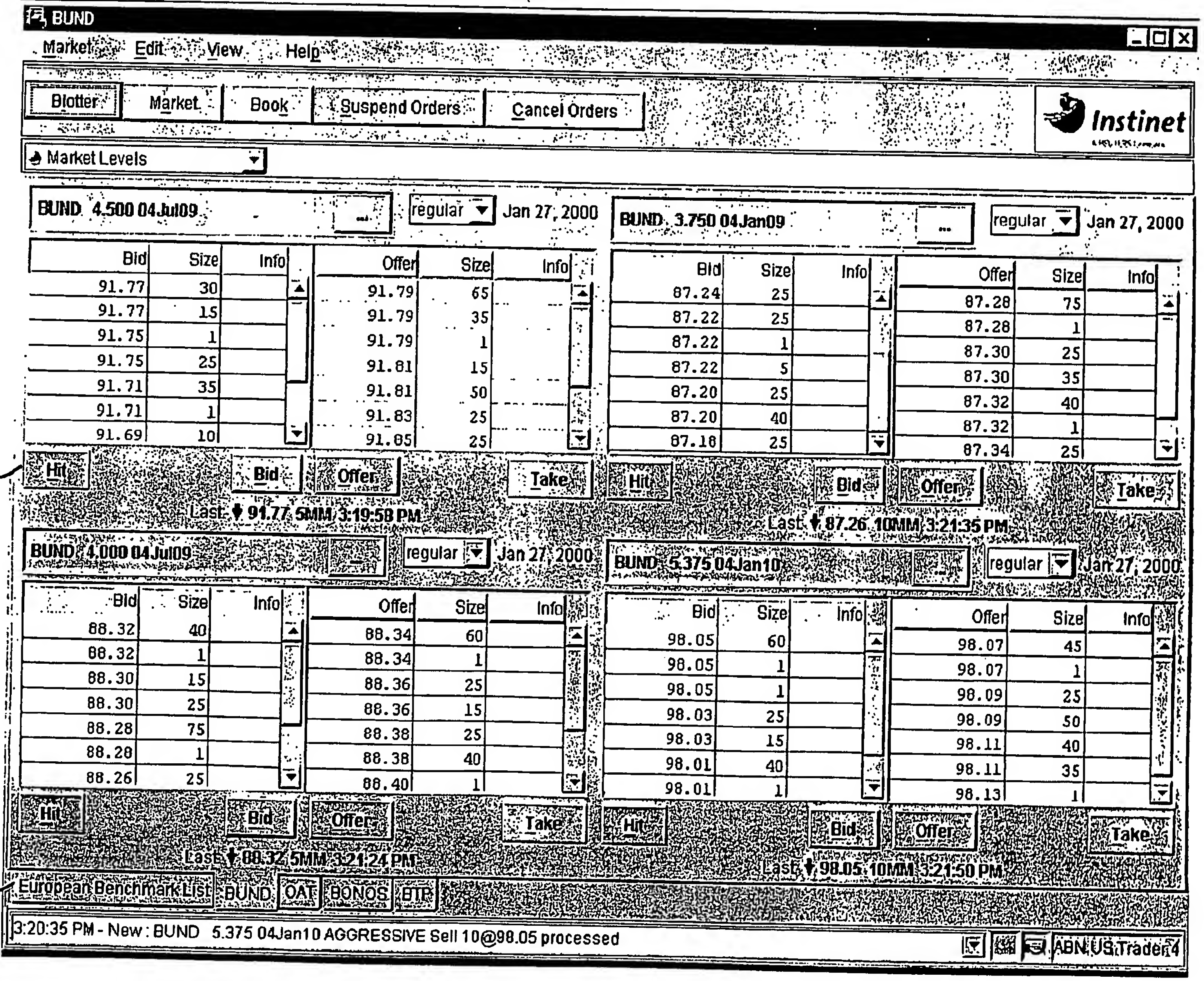
Figure 38b



```
4:16:15 PM - UST   5.750 30Jun01 5@99.49609375 - bought
4:15:26 PM - UST   5.750 30Jun01 1@99.49218750 - sold
```

Figure 39

Figure 4D



BUND 4.500 04Jul09 regular Jan 27, 2000

Bid	Size	Info	Offer	Size	Info
91.77	30		91.79	65	
91.77	15		91.79	35	
91.75	1		91.79	1	
91.75	25		91.81	15	
91.71	35		91.81	50	
91.71	1		91.83	25	
91.69	10		91.85	25	

Last 91.77 5MM 3:19:58 PM

BUND 3.750 04Jan09 regular Jan 27, 2000

Bid	Size	Info	Offer	Size	Info
87.24	25		87.28	75	
87.22	25		87.28	1	
87.22	1		87.30	25	
87.22	5		87.30	35	
87.20	25		87.32	40	
87.20	40		87.32	1	
87.18	25		87.34	25	

Last 87.26 10MM 3:21:35 PM

BUND 4.000 04Jul09 regular Jan 27, 2000

Bid	Size	Info	Offer	Size	Info
88.32	40		88.34	60	
88.32	1		88.34	1	
88.30	15		88.36	25	
88.30	25		88.36	15	
88.28	75		88.38	25	
88.28	1		88.38	40	
88.26	25		88.40	1	

Last 88.32 5MM 3:21:24 PM

BUND 5.375 04Jan10 regular Jan 27, 2000

Bid	Size	Info	Offer	Size	Info
98.05	60		98.07	45	
98.05	1		98.07	1	
98.05	1		98.09	25	
98.03	25		98.09	50	
98.03	15		98.11	40	
98.01	40		98.11	35	
98.01	1		98.13	1	

Last 98.05 10MM 3:21:50 PM

3:20:35 PM - New : BUND 5.375 04Jan10 AGGRESSIVE Sell 10@98.05 processed

Figure 41

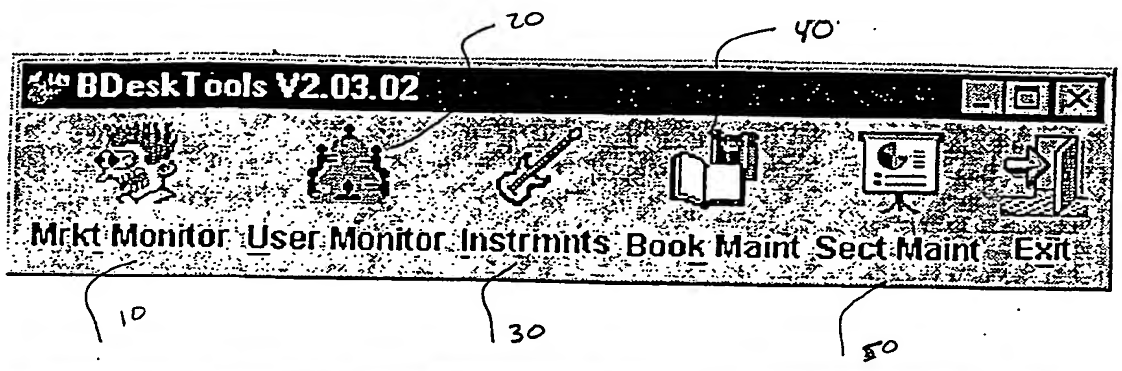


Figure 44

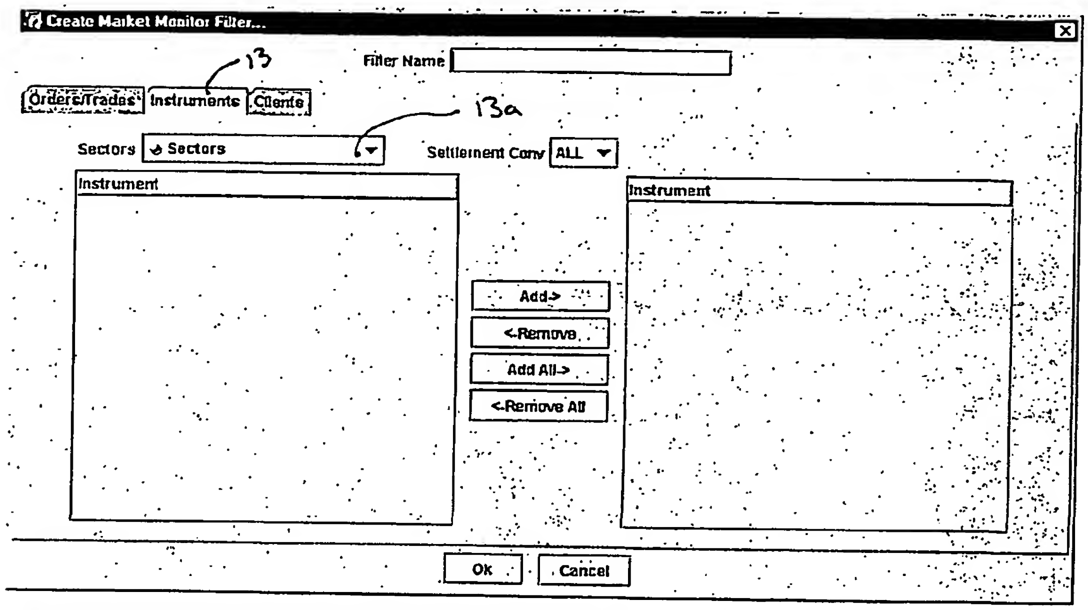
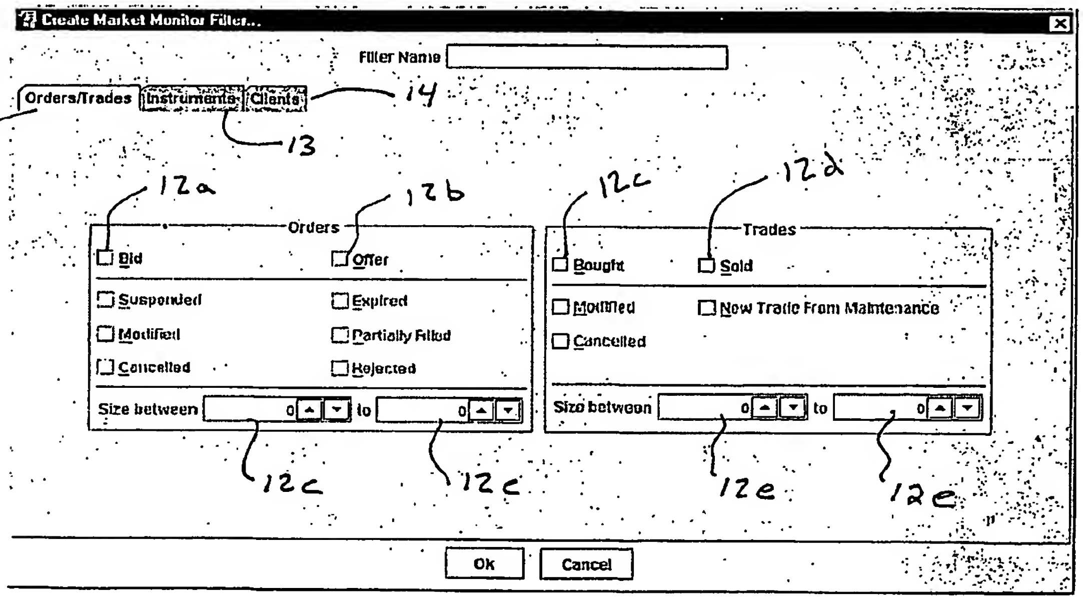
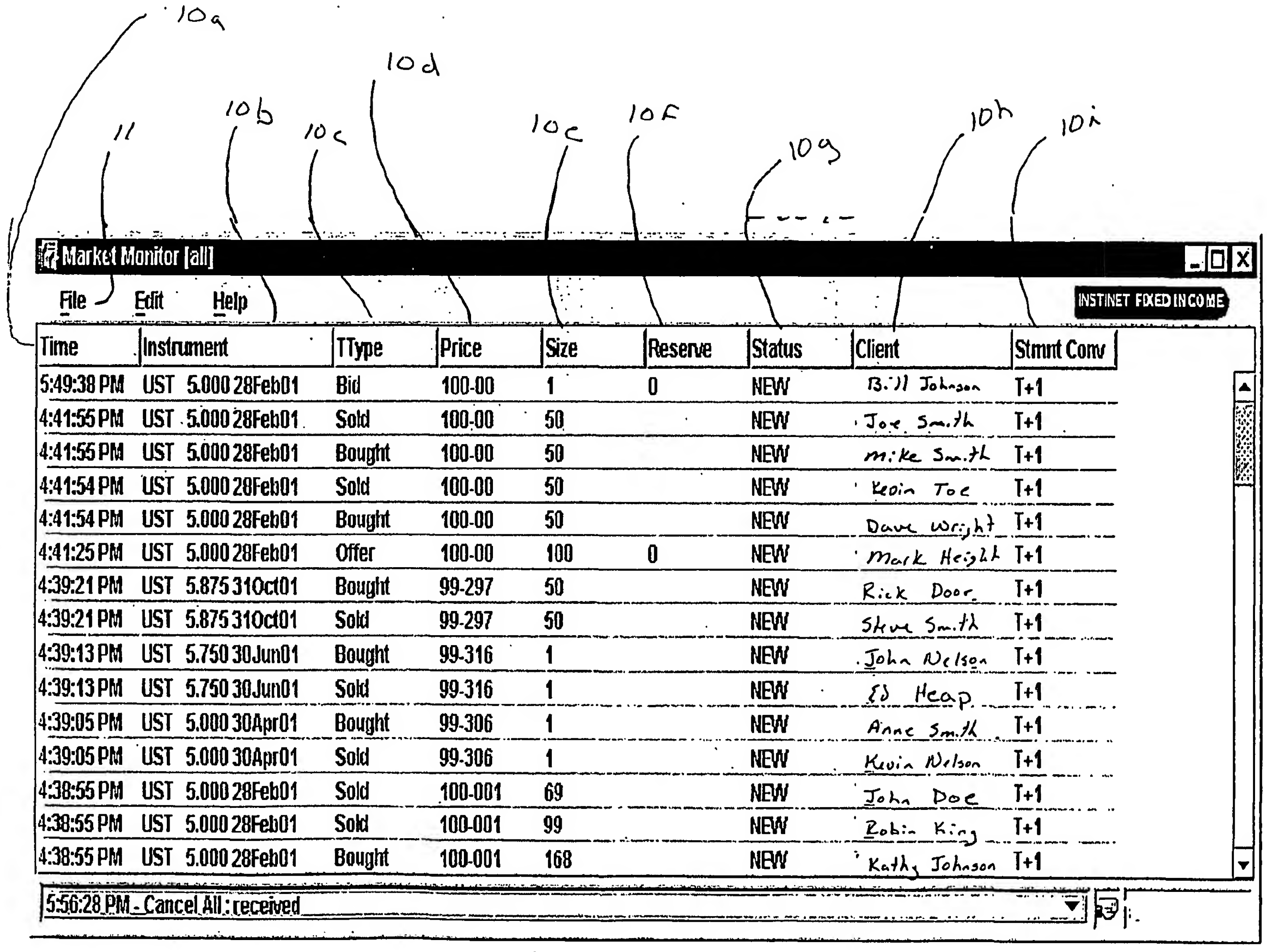


Figure 43

12





Time	Instrument	TType	Price	Size	Reserve	Status	Client	Stmnt Conv
5:49:38 PM	UST 5.000 28Feb01	Bid	100-00	1	0	NEW	Bill Johnson	T+1
4:41:55 PM	UST 5.000 28Feb01	Sold	100-00	50		NEW	Joe Smith	T+1
4:41:55 PM	UST 5.000 28Feb01	Bought	100-00	50		NEW	mike Smith	T+1
4:41:54 PM	UST 5.000 28Feb01	Sold	100-00	50		NEW	Kevin Toe	T+1
4:41:54 PM	UST 5.000 28Feb01	Bought	100-00	50		NEW	Dave Wright	T+1
4:41:25 PM	UST 5.000 28Feb01	Offer	100-00	100	0	NEW	Mark Height	T+1
4:39:21 PM	UST 5.875 31Oct01	Bought	99-297	50		NEW	Rick Door	T+1
4:39:21 PM	UST 5.875 31Oct01	Sold	99-297	50		NEW	Steve Smith	T+1
4:39:13 PM	UST 5.750 30Jun01	Bought	99-316	1		NEW	John Nelson	T+1
4:39:13 PM	UST 5.750 30Jun01	Sold	99-316	1		NEW	Ed Heap	T+1
4:39:05 PM	UST 5.000 30Apr01	Bought	99-306	1		NEW	Anne Smith	T+1
4:39:05 PM	UST 5.000 30Apr01	Sold	99-306	1		NEW	Kevin Nelson	T+1
4:38:55 PM	UST 5.000 28Feb01	Sold	100-001	69		NEW	John Doe	T+1
4:38:55 PM	UST 5.000 28Feb01	Sold	100-001	99		NEW	Robin King	T+1
4:38:55 PM	UST 5.000 28Feb01	Bought	100-001	168		NEW	Kathy Johnson	T+1

5:56:28 PM - Cancel All: received

Figure 42

Figure 45

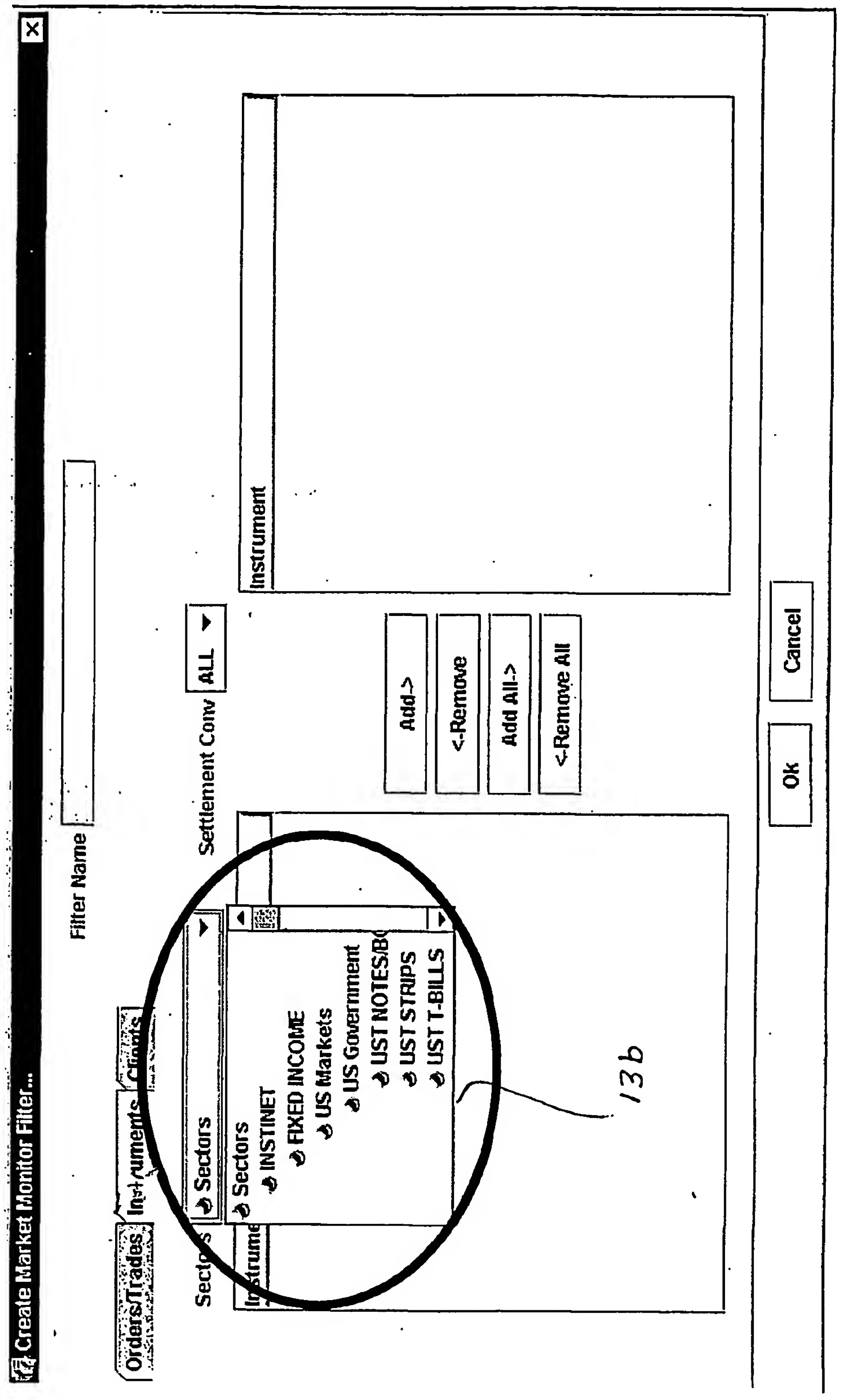
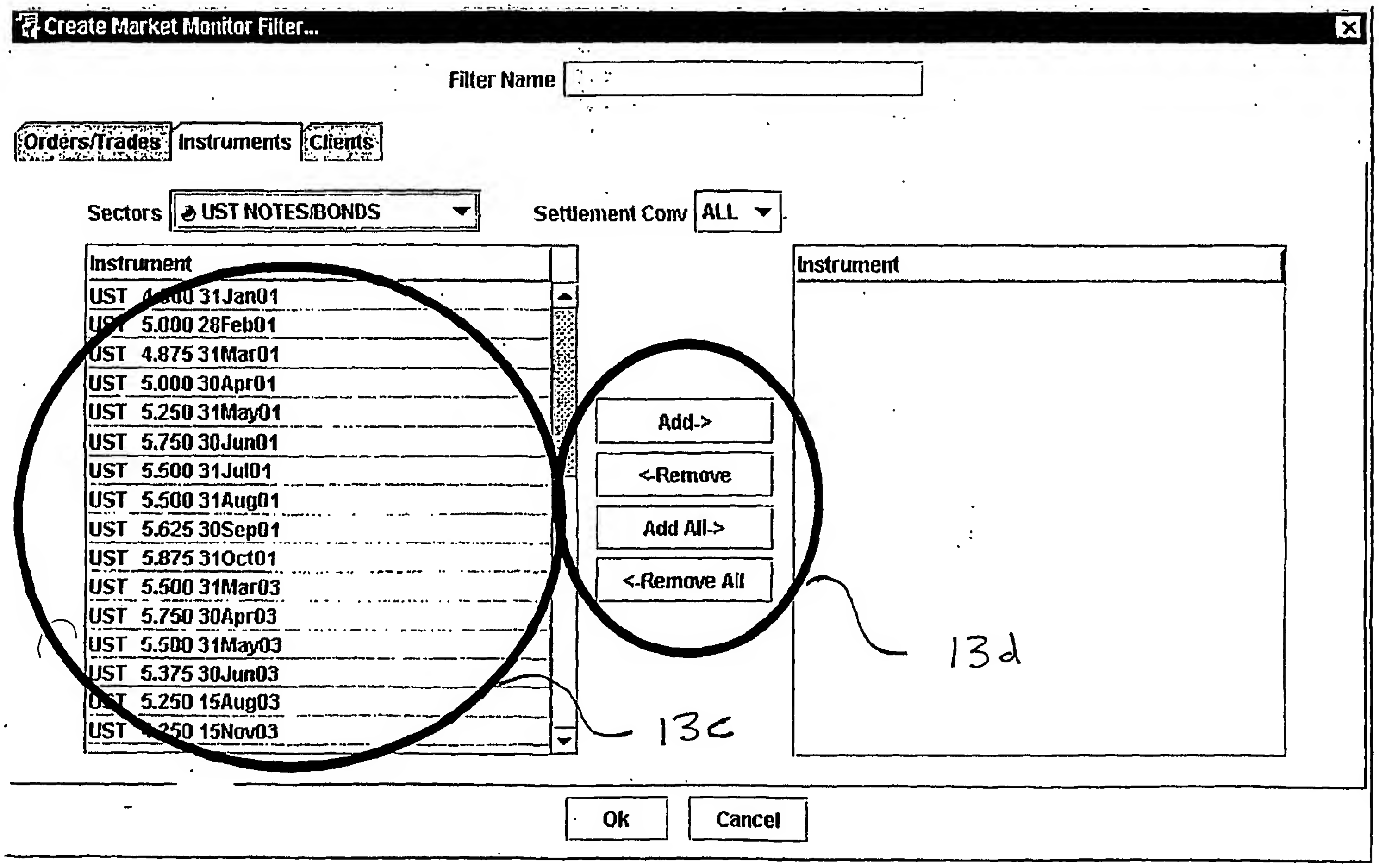

Create Market Monitor Filter...
Filter Name
Orders/Trades
Instruments
Clients
Sectors
Sectors
Sectors
INSTINET
FIXED INCOME
US Markets
US Government
UST NOTES/BO
UST STRIPS
UST T-BILLS
Instrume
Settlement Conv
ALL
Instrument
Add->
<-Remove
Add All->
<-Remove All
Ok
Cancel
13b

Figure 46



Filter Name

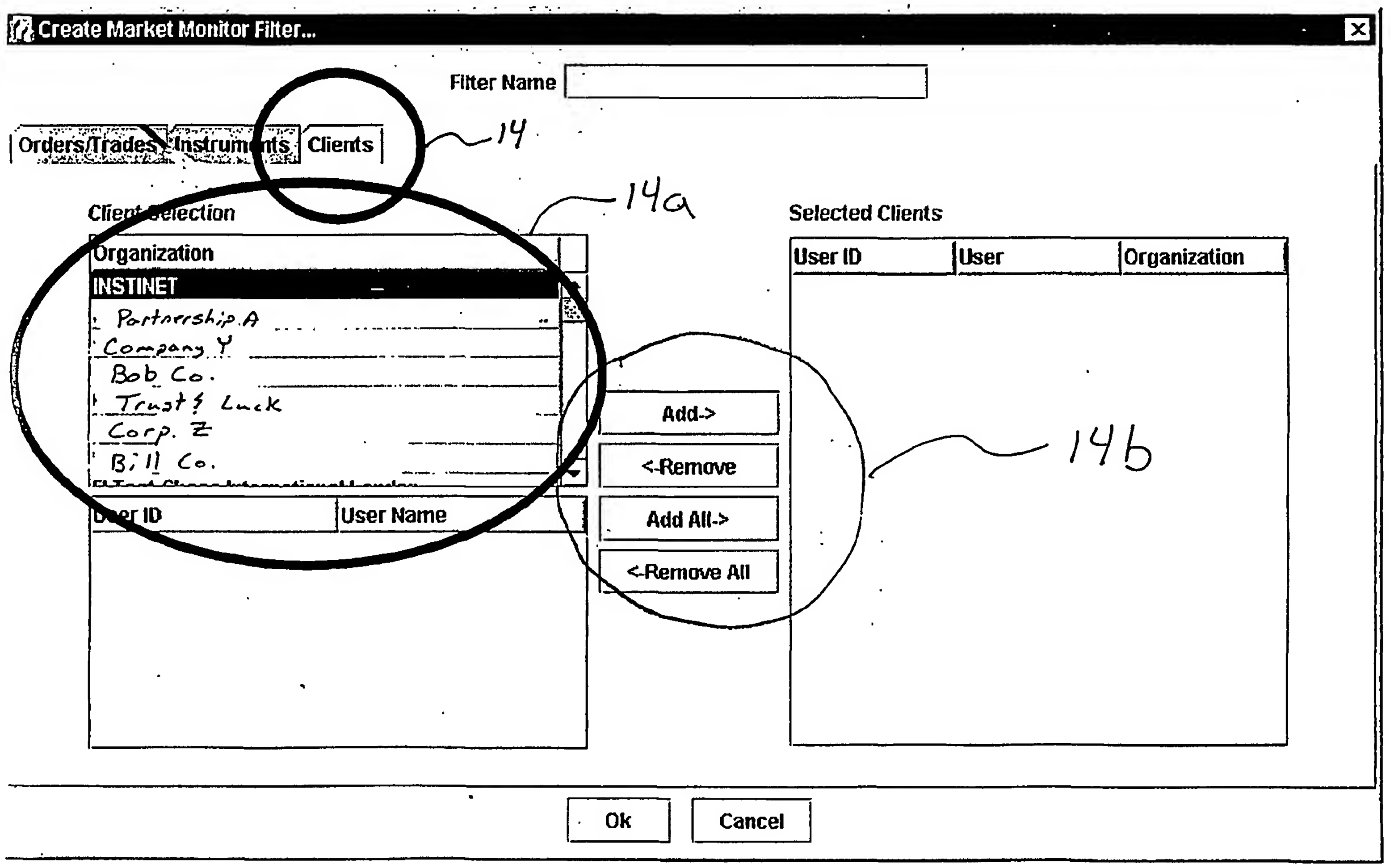
Sectors: UST NOTES/BONDS

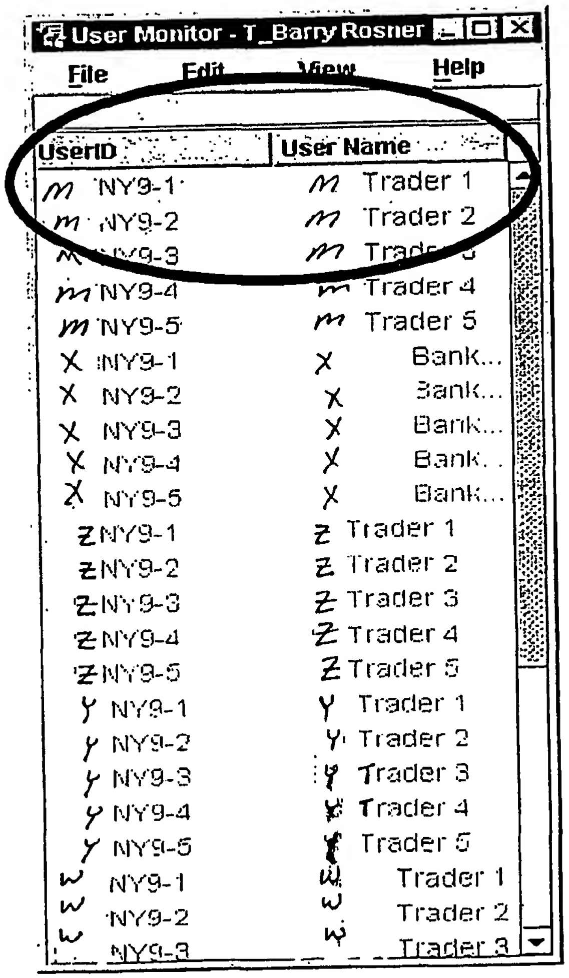
Settlement Conv: ALL

Instrument
UST [illegible] 31Jan01
UST 5.000 28Feb01
UST 4.875 31Mar01
UST 5.000 30Apr01
UST 5.250 31May01
UST 5.750 30Jun01
UST 5.500 31Jul01
UST 5.500 31Aug01
UST 5.625 30Sep01
UST 5.875 31Oct01
UST 5.500 31Mar03
UST 5.750 30Apr03
UST 5.500 31May03
UST 5.375 30Jun03
UST 5.250 15Aug03
UST [illegible]250 15Nov03

Instrument

Figure 47

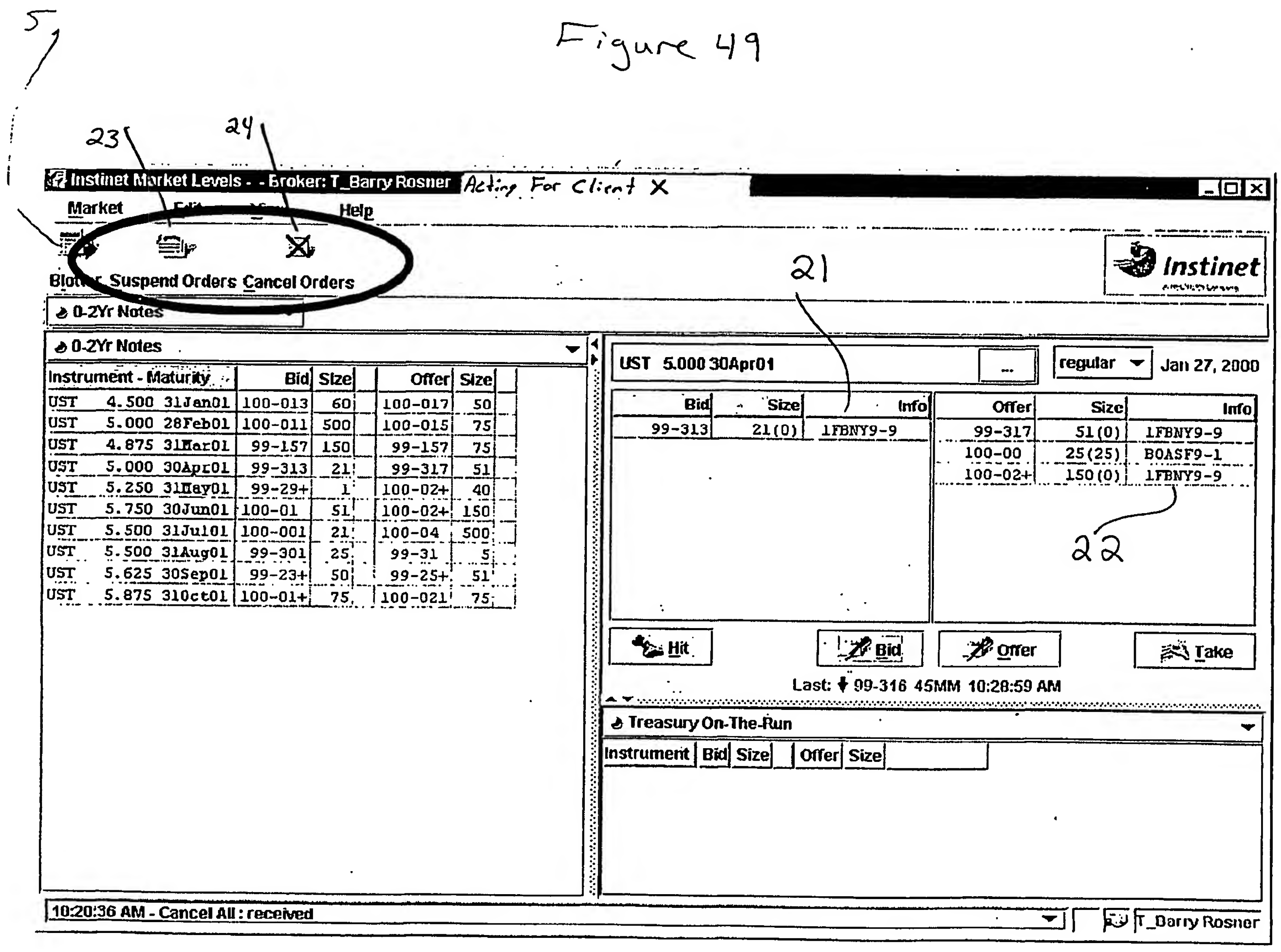

Create Market Monitor Filter...
Filter Name
Orders/Trades
Instruments
Clients
14
14a
14b
Client Selection
Organization
INSTINET
Partnership A
Company Y
Bob Co.
Trust $ Luck
Corp. Z
Bill Co.
User ID
User Name
Add->
<-Remove
Add All->
<-Remove All
Selected Clients
User ID
User
Organization
Ok
Cancel



UserID	User Name
m NY9-1	m Trader 1
m NY9-2	m Trader 2
m NY9-3	m Trad
m NY9-4	m Trader 4
m NY9-5	m Trader 5
x NY9-1	x Bank...
x NY9-2	x Bank...
x NY9-3	x Bank...
x NY9-4	x Bank...
x NY9-5	x Bank...
z NY9-1	z Trader 1
z NY9-2	z Trader 2
z NY9-3	z Trader 3
z NY9-4	z Trader 4
z NY9-5	z Trader 5
y NY9-1	y Trader 1
y NY9-2	y Trader 2
y NY9-3	y Trader 3
y NY9-4	y Trader 4
y NY9-5	y Trader 5
w NY9-1	w Trader 1
w NY9-2	w Trader 2
w NY9-3	w Trader 3

FIGURE 48

Figure 49



Instrument - Maturity	Bid	Size	Offer	Size
UST 4.500 31Jan01	100-013	60	100-017	50
UST 5.000 28Feb01	100-011	500	100-015	75
UST 4.875 31Mar01	99-157	150	99-157	75
UST 5.000 30Apr01	99-313	21	99-317	51
UST 5.250 31May01	99-29+	1	100-02+	40
UST 5.750 30Jun01	100-01	51	100-02+	150
UST 5.500 31Jul01	100-001	21	100-04	500
UST 5.500 31Aug01	99-301	25	99-31	5
UST 5.625 30Sep01	99-23+	50	99-25+	51
UST 5.875 31Oct01	100-01+	75	100-021	75

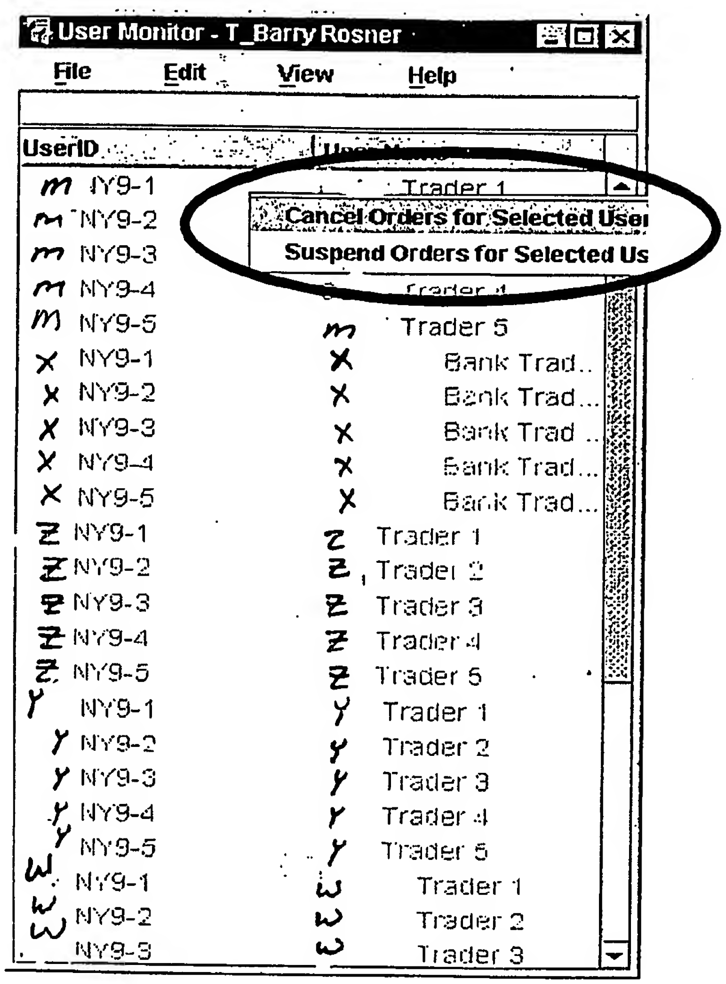
Bid	Size	Info
99-313	21(0)	1FBNY9-9

Offer	Size	Info
99-317	51(0)	1FBNY9-9
100-00	25(25)	BOASF9-1
100-02+	150(0)	1FBNY9-9

Last: ↓ 99-316 45MM 10:28:59 AM

Instrument	Bid	Size	Offer	Size

10:20:36 AM - Cancel All : received


User Monitor - T_Barry Rosner
File
Edit
View
Help
Cancel Orders for Selected User
Suspend Orders for Selected Us

UserID	[illegible]
m NY9-1	Trader 1
m NY9-2	
m NY9-3	
m NY9-4	[illegible]
m NY9-5	m Trader 5
x NY9-1	x Bank Trad...
x NY9-2	x Bank Trad...
x NY9-3	x Bank Trad ...
x NY9-4	x Bank Trad...
x NY9-5	x Bank Trad...
z NY9-1	z Trader 1
z NY9-2	z Trader 2
z NY9-3	z Trader 3
z NY9-4	z Trader 4
z NY9-5	z Trader 5
y NY9-1	y Trader 1
y NY9-2	y Trader 2
y NY9-3	y Trader 3
y NY9-4	y Trader 4
y NY9-5	y Trader 5
w NY9-1	w Trader 1
w NY9-2	w Trader 2
w NY9-3	w Trader 3

FIGURE 50